universität wien

MASTERARBEIT / MASTER'S THESIS

Titel der Masterarbeit / Title of the Master's Thesis

„Identification of Communication Patterns in Network Flow Data using Social Network Analysis“

verfasst von / submitted by

Alexandra-Madalina Tamas, BSc.

angestrebter akademischer Grad / in partial fulfilment of the requirements for the degree of

Diplom-Ingenieur (Dipl.-Ing.)

Wien, 2018 / Vienna 2018

Studienkennzahl lt. Studienblatt / degree programme code as it appears on the student record sheet:	A 066 926
Studienrichtung lt. Studienblatt / degree programme as it appears on the student record sheet:	Masterstudium Wirtschaftsinformatik
Betreut von / Supervisor:	Univ.-Prof. Dipl.-Math. Dr. Stefanie Rinderle-Ma

Declaration of Authorship

I, Alexandra-Madalina Tamas, BSc, declare that this thesis titled, “Identification of Communication Patterns in Network Flow Data using Social Media Analysis” and the work presented in it are my own. I confirm that:

- This work was done wholly or mainly while in candidature for a research degree at this University.
- Where any part of this thesis has previously been submitted for a degree or any other qualification at this University or any other institution, this has been clearly stated.
- Where I have consulted the published work of others, this is always clearly attributed.
- Where I have quoted from the work of others, the source is always given. With the exception of such quotations, this thesis is entirely my own work.
- I have acknowledged all main sources of help.
- Where the thesis is based on work done by myself jointly with others, I have made clear exactly what was done by others and what I have contributed myself.

Signed:

_____

Date:

_____

Acknowledgments

I want to offer my gratitude and appreciation to Ms. Stefanie Rinderle-Ma for her permanent support and guidance throughout the completion of this thesis.

Thanks to Ma Zhendong for the inspiring talks and for his assistance.

My deepest thankfulness goes to Giovanna Stanciu, my role model, who is the living proof that only by hard work and perseverance you can succeed and make your dreams come true.

I would like to express gratitude to my friends, who encouraged me and always have been there for me.

Finally yet importantly, I want to thank to my family, for believing in me and for the unconditional support and encouragement, despite the hundreds of kilometers that separate us.

Table of Contents

List of Figures

List of Tables

1 Motivation

Computer networks play a massive role in our daily lives. From social media to Internet surfing or streaming, they enable most of the activities that we do today. Their applicability extends beyond the usage at home/for private matters up to enterprise level. Both communication and information are valuable strategic concerns for assuring the success of every corporation. To overcome the obstacle of connecting departments between each other or to data recovery programs by effectively using information technology, it is necessary to organize computer systems with the intent of merging both machines and communications. This type of organization encapsulates the concept of computer networking. By using computer networks the differentiation between devices that store and process information and those that collect and transport it disappears and the obstruction between data stored on various systems is removed, as well.

Synchronously with the appearance and development of computer systems, and implicitly of the crucial personal and business data being stored on them, concerns related to their security arose, as well as the need for understanding what is behind the communication processes. Technologies for assuring network security improve and develop constantly, since the attack methods become also more refined. Beyond physical network security, which is mainly concerned with protecting the hardware against physical intrusion, password protection, installing anti-spyware software or using VPN services, which are all prevention methods against attacks, it is interesting to analyze also how network assets communicate with each other, in order to not only prevent, but also address already possible existing issues.

Both security architects and network engineers have wondered about how to understand better the communication in computer networks, in order to further optimize, replace or discard existing services. As with this need for understanding, several tools have been developed, which provide useful information and statistics about the network traffic of interest. Some of the network analysis tools such as Wireshark[1], Zenoss Core[2], Network Miner[3] or Zenmap[4] can be used for troubleshooting complex problems, identifying malicious activity (malware) and unused protocols, generating traffic for penetration testing or working with an Intrusion Detection System (IDS)[5].

---

[1] https://www.wireshark.org/

[2] https://www.zenoss.com/

[3] https://www.netresec.com/?page=NetworkMiner

[4] https://nmap.org/zenmap/

[5] https://en.wikipedia.org/wiki/Intrusion_detection_system

A detailed analysis of the network structure and behavior can be accomplished by analyzing the underlying communication patterns of the network traces by means of social network analysis (SNA)[6], which is both a way of theorizing the social structure and its consequence on behavior, and a method for analyzing the volume and patterns of social relations between individual actors [33, p.31]. A communication pattern can be defined as a regular form or motif identified by a set of statistical measures applied on the network data, which is valid also for other similar network data sets that is applied on. In other words, if a communication pattern is identified within network A, then there should be a high probability that this pattern is found also in network B, under certain conditions. Finding communication patterns is important for describing and anticipating the behavior of hosts within a data network. These patterns may correspond to both attack activities and activities concerning hosts having key roles within the network traffic (for instance, hosts sending a very large amount of bytes or hosts whose removal disconnect the network).

Social network behavior refers to whom is talking to whom, how much they are communicating and the patterns of this interaction, as well. For instance, it is appropriate to ask also which host (a computer or other device) communicates to which host, how many bytes do these hosts transfer between each other and what are the patterns within the network traces. A network trace contains information about what data is transferred when connecting a device to the network. All network traffic is derived from social networking, so it is be possible to evaluate social network behavior from trace files. The main aspect that differentiates social network behavior from conventional traffic theory is that the first focuses on end-to-end patterns, while the second emphasizes the traffic behavior over time. For instance, a flow is viewed in the conventional traffic theory as a generalization of a TCP flow, so this flow has a start and an end. As opposed to this, SNA does not focus on the beginning and on the termination of a flow, but on the overall traffic between two endpoints, which is considered a network flow [45, p.3]. The analysis of network flow information is more effective than deep packet inspection since both internal (e.g. contravention of the policies) and external activities (e.g. intrusions) are considered and the user privacy is covered [45, p.2].

Although attempts have been made to characterize network flow traffic based on social network behavior, none of these attempts aimed to compare the findings resulted from applying the conventional methods of network profiling (these methods are based on the given attributes of the network such as port[7] or protocol[8]), and the results obtained by using SNA on the network of interest. In addition, no effort has been

---

[6] https://en.wikipedia.org/wiki/Social_network_analysis

[7] https://en.wikipedia.org/wiki/Port_(computer_networking)

[8] https://searchnetworking.techtarget.com/definition/protocol

made so far to analyze the communication patterns between other attributes of the network except the IP addresses (which will be referred further as IPs) and the relationships between them. For instance, it is also meaningful to analyze the (IP, port) pair as a whole and see how it connects to other (IP, port) pairs.

In this thesis, a substitute path is examined, by first profiling the network using and partly also optimizing conventional methods and then applying SNA measures on two data sets, in order to find correlations between the first and the latter findings. The first data set is analyzed in an explorative manner and the findings resulted from this analysis are evaluated by means of the second data set. The focus is set on identifying anomalous traffic, but also peer-to-peer (P2P[9]) flows (which are, by nature, insecure), since the conventional methods for identifying the hosts involved in P2P interactions are quite complex and require a high computational effort. The analysis can be applied on further network data sets as well, by executing a self-configurable script. The results of this thesis bring us one step closer to answering the question if only analyzing the dynamics of the network by means of SNA is sufficient for identifying the most influential assets and possible anomalies.

1.1 Problem statement

The tools used for modern network management identify heavy-hitters by focusing on traffic volume. Even if these tools support many statistics and visualizations of the network and can be used for both passive and active network measurements, they do not provide an overview of the structure contained in network data. The statistics supported by these tools focus primarily on the protocols and ports used and on the bytes transferred between the hosts. Wireshark is an appropriate tool to view the network usage by protocol or to see all the traffic visible of one specific interface, although the notifications received by the user do not confirm the presence of a possible anomaly within the network. If the user is interested in an easier troubleshooting and configuration of the network, then Cisco Prime[10] is a good choice, which contains also real-time information about areas in the network. Nevertheless, Cisco Prime is proprietary and may be expensive. Zenmap performs fast DNS lookups and scans the network for possible exposures, but its complexity makes it hard for a non-experienced user to get a detailed overview of the network characteristics, in order to detect anomalies and/or patterns. NetFlow[11] provides traffic statistics concerning user, port, protocol and type of services and supports data analyzing and visualizing. Thus, none of these software and other alike provide:

---

[9] https://en.wikipedia.org/wiki/P2P

[10] https://www.cisco.com/c/en/us/products/cloud-systems-management/prime.html

[11] https://www.cisco.com/c/en/us/products/ios-nx-os-software/ios-NetFlow/index.html

- A detailed characterization of each flow, which might be useful for finding correlations among the attributes of the network.
- A reliable profiling of the exposed traffic (for instance P2P) since the identification of applications based on port numbers is becoming more and more inaccurate in classifying anomalous traffic.
- Adequate dimensionality.

The mentioned tools provide statistics based on the protocol or the ports used within the network of interest. Similar statistics and also other additional insights can be acquired when profiling the network by using the SNA methods. Some publications address the SNA topic with regard to network traffic. The approach addressed in [1] uses the social network analysis theory in a P2P network and introduces an algorithm based on the topology potential theory to separate the network and search for core nodes. This simplifies the monitoring of P2P networks, by analyzing only the center nodes. [2] addresses the identification of the typical features of the fundamental social network derived from the traffic using methods of frequency analysis and traffic matrices. [3] suggests a community detection algorithm, which is based on one mode projection and bipartite networks. This approach uses graph partitioning of the similarity graph for discovering communities in large-scale networks. These studies use SNA for generating meaningful outcomes, still none of them takes into consideration the entire network and the correlation between all of its attributes, which can reveal some interesting patterns. While [1] searches in distinct subgroups of the network for patterns, [2] takes the interpretation of one link of the traffic as a guide for understanding the behavior of the whole network and [3] relies exclusively on the IP connectivity for finding clusters, without any information about the protocol, packets being exchanged or used ports.

Another problem is the visualization concept of these methods, which focuses rather on particular structural elements of the network than on emphasizing the communication patterns between these elements.

1.2 Research Question

As shown in Section 1.1, no study was concerned with profiling all dimensions of the network and with the examination of the correlations between all the statistical results obtained by using SNA. Thus, this thesis is created upon one main research question, which is further divided into two sub questions:

RQ: What is the extent to which the measures of SNA can be used to find communication patterns within the network and what is the meaning of these patterns?

Following sub questions extend the main question:

RQ/SQ1: Which communication patterns within a network of interest are determined by the given attributes of the network?

RQ/SQ2: Which network assets require a closer investigation based on the results of the applied methodology?

When considering these aspects, important patterns of communications can be found, which can be further used for optimizing the network performance.

Therefore, the results of this thesis are the identification of common services, P2P traffic and anomalous activity, as described in Section 3.6, Section 3.7 and Section 3.8, and the methods of SNA outlined in Section 3.9, which supplement or confirm the first findings. These two main approaches lead to the ascertainment of communication patterns within the network of interest. The network's attributes are, among others, the source- and destination IP address, the source- and destination port number, the protocol, the transfer bytes and the number of packets. These attributes are also the ones that are used in a first phase to determine the well-known services, the P2P traffic and a possible anomalous activity. The second phase of the chosen methodology aims to determine which measures and visualization techniques of SNA are assigned to which type of network behavior and structure identified in the first phase and how these results can enhance or replace the first findings.

It has to be awaited, that the chosen visualization techniques complement the statistical methods used and provide a guidance on which network assets need to be further monitored. For a more detailed insight into the network structure, it is recommended to use tools that are created mainly for the purpose of visualization such as Gephi[12], Cytoscape[13] or Graphviz[14].

1.3 Contribution

This thesis provides a novel approach for identifying communication patterns within network traces, by means of SNA. The main contributions of this thesis are the following:

– Most existing studies apply only particular measures of SNA on the network. This thesis covers the vast majority of these measures and the analysis results are mapped on flow level, in order to cover all dimensions of the network and outline communication rules that govern most of the user activities existing in the trace. These methods are applied beyond rules involving only the IP addresses as significant units of relevance for the analysis and extend up to (port, port), (IP, port), (pair, IP) and (pair, Port) pairs analysis, where a pair is defined by an (IP, port) tuple. Such an expansive picture provides a

[12] https://gephi.org/

[13] http://www.cytoscape.org/

[14] https://www.graphviz.org/

network administrator/operator with an examination of the reality: it can be noticed how traffic assigned to leading applications crosses over the network and configuration errors, not reasonable communication and suspicious activity can be detected.
- The identified P2P traffic is analyzed with respect to its embedment within the network clusters, providing insight into what network components require a closer investigation or monitoring.
- The identified common services are analyzed with respect to the betweenness, closeness and degree centrality and to their affiliation to the egocentric networks and network components. Thus, the main servers and clients can be identified by studying only the network dynamics, without relying on the information existent in the IP header fields.
- Analyzing data in terms of equivalence classes enables the generalizations about social structure and behavior, by yielding which host can be substituted by another host that has the same links to a group of hosts, respectively which hosts have the same positions within the network with respect to all other hosts.
- The development of betweenness, degree, neighbors and bytes over time gives insight into the most critical time span(s). By considering the peaks (local maxima) of the temporal plots, a further investigation of the assets that are responsible for an unusual traffic activity is possible.
- A user applying the applied methodology to profile a network gets an overview of the most exposed and/or influential assets including their assigned ports, the possible anomalies and the P2P traffic, all mapped both on node and on flow level.
- The results of this analysis are summarized in form of a HTML report, which provides guidance on which assets of the network need to be investigated and/or monitored;
- The R script that generates the analysis results can be manually configured by the user, in order to meet the desired requests. The script can be executed repeatedly on various network data in order to learn dependencies for the applications being used.

1.4 Structure

This thesis is structured as follows: Section 2 outlines the related work, by summarizing the SNA methods addressed in the existing publications, which are applied on network data. Section 3 provides an accurate characterization of the methodology applied on both data set I and data set II and includes the state of the art survey, the description of the data sets and the way they are transformed and processed. This preliminary step of data preparation is followed by describing the implementation of the methods used for identifying common services, anomalous activity and P2P traffic and the visualization of their underlying patterns, including a summary of the chosen

SNA statistical and visualization methods. The methodology defined in Section 3 is applied in the first phase in Section 4, which contains the explorative analysis of the first data set. The outcome of the methods applied in Section 4 are summarized in form of conceptual propositions in Section 5. The same methods applied on the first data set in Section 4 are applied in the second phase on another data set in Section 6, with the intent of validating the conceptual propositions formulated. The observations resulted from the analysis, the limitations of the methods and the possible challenges are summarized in Section 7 and the conclusion and the summary are presented in Section 8. The other sections are necessary for covering formal features of the thesis but they do not include any new observations related to the research questions. The R script generates a HTML report, which contains some of the plots and the data frames that enable the understanding of the network behavior and structure. A data frame is the fundamental data structure of the R software and consists of rows (describing the actors) and columns (describing the attributes of these actors) [13, p.79].

2 State of the Art

Most of the previous work has focused on particular aspects of the network traffic. For example, [6] presents a general algorithm for detecting flow clusters within a packet trace that are temporally correlated. For this purpose, they developed the tool eXpose, which focuses on the flow pairs that are statistically significant, by defining a concept of common rules. The tool can reveal possible configuration errors and provides an overview of the activity within the network without having to analyze the server logs. Another study concerned with the feature estimation of the fundamental social network from the whole traffic of interest is [2], which introduces methods based on frequency analysis and traffic matrices. The analysis takes into consideration the most predominant origin-destination flow pairs and seeks to identify the main statistical characteristics, which describe the network behaviour with regard to the origins, destinations, and the origin-destination flows. [7] proposes the use of Traffic Dispersion Graphs (TDGs) for analysing and visualizing network traffic. These graphs model the social behaviour of hosts, where the interactions between these hosts can have different semantics (for instance, the number of packets or the type of packets being transferred). The identification of applications from a given graph is not supported, neither the consideration of edge weights for analysing the frequency of an edge. The social behaviour similarity of end-hosts within the same network prefixes is addressed in [8], by using bipartite graphs for modelling the traffic and one-mode projection graphs for assessing the similarity. A spectral clustering algorithm is used for grouping the hosts within same prefixes into distinct behaviour clusters. The aim of this algorithm is the detection of anomalous activity by using a synthetic traffic that combines both real flows containing attacks against the network and Internet backbone traffic. The algorithm suggested in [9] detects community structures in networks

and is based on the grade of the node membership and the integration of sub-communities. The method has a low time complexity since it depends mainly on the local information of each host. [10] introduces a clustering algorithm which analyses the most critical IP addresses, by using traces with a long duration. According to [10], analysing the traffic over a long time supports a more constant host behaviour and clustering the hosts leads to the emergence of models describing a valid Internet communication. Similar to [10], an explanatory analysis of cluster behaviours was proposed also by [11], which focuses on the identification of recognized behaviour profiles and how they support the detection of anomalous or suspicious activity. [18] uses data mining and entropy-based methods to profile valuable information within large networks by focusing on cluster extraction.

In contrast, this thesis aims to build behaviour profiles at host level and then extend them to network level. The findings are assigned on flow level within a data frame, in order to cover the whole extent of the network and define communication patterns that rule the behavior of the assets.

3 Methodology

The process of identifying communication patterns in data sets begins with an attribute-based analysis for detecting well common services, P2P traffic and suspicious activity. Subsequently, the structure of connections to which the actors belong is examined. This approach represents the basis of SNA, which focuses on the relations between actors and not on their attributes [13, p.3]. The analysis of social relationships reveals the differences between the network positions of the actors and how these network positions constrain the actors' behaviour [13, p.10]. For this purpose, a rather statistical than mathematical approach is adopted. The difference between the two approaches is that the first one regards the data as a sample of a larger population and assumes that the results are reproducible for other similar network data, while the second one treats the observations as the population of interest [13, p.15]. Both attribute- and relationship-based approaches provide valuable insight into the structure and behaviour of the network. This thesis follows a top-down perspective, which aims to characterize whole populations by the relations patterns that restrain its actors.

3.1 Research Strategy

The first step made after being provided with the first data set from Austrian Institute of Technology (AIT)[15] was to explore the documentation of the tool used to collect

[15] https://www.ait.ac.at/en/

the data (NFdump[16]). From here, other tools (which are outlined in Section 1.1 and Section 1.2) used to process and analyse this type of data were searched and their functionalities were compared. In parallel with this step, the topic of network traffic analysis was deepened, by reading papers and publications concerned with both aspects of network security and network behaviour. Relevant here was the literature that uses SNA to explain the network behaviour and not the papers focused exclusively on the description of new software designed for monitoring or improving network behaviour, since they do not contain a detailed insight into the network analysis and do not fit to the scope of this research. The latter findings were analysed, compared and evaluated including their deficiencies and needs of improvement. This evaluation led to the choice of analysing the network across all its dimensions, by assigning different meanings to its elements and to the way in which they are interconnected.

3.2 Data Collection and Description

The first data set provided by Austrian Institute of Technology was collected from the perimeter of an enterprise network. The switches and routers used to collect the data monitored the bidirectional traffic for 225 local hosts, which form the focal point of the further analysis. The data was anonymized after collection to protect the user privacy without hindering its utility. NetFlow is a protocol developed by Cisco[17] in order to describe network operations and to facilitate the understanding of network behaviour. The data was read from the network and stored by the NetFlow capture daemon of the nfdump tools, which can both collect and process NetFlow data on the command line.

So that the data can be imported and analysed in RStudio[18], the binary format of the NFCAPD file has to be transformed into a CSV file for further use, by using a Linux based system. It was opted for Ubuntu 16.04 LTS. This results in a CSV file with 48 columns and 13264 rows. Each row represents one unidirectional flow and each column describes an attribute. Only 11 out of 48 columns contain values that are not zero or not blank. These columns include the start- and end time (“ts”, ”te”), the source and destination address (“sa”, “da”) , the source and destination port (“sp”, “dp”), the protocol (“pr”), the source type of service (“stos”), the input packets (“ipkt”), the input bytes (“ibyt”) and the input interface number (“in”).
These dimensions are the ones that are evaluated. The columns assigned to the start and end time are identical, so it can be assumed that one row describes one unidirec-

[16] http://nfdump.sourceforge.net/

[17] https://www.cisco.com/

[18] https://www.rstudio.com/

tional flow, since it is not possible that all transfers occur within an interval of 0 seconds. The aggregation of these unidirectional flows is described in Section 3.4. All the flows define the network traffic between 10:59:49 and 11:05:01, which corresponds to a time interval of 5 minutes and 12 seconds. The traces contain mostly TCP and UDP traffic except for a few ICMP and ESP traffic.

The columns that contain only zero values or no values at all are duration ("td"), TCP flags ("flg"), forwarding status ("fwd"), output packets ("opkt"), output bytes ("obyt"), output interface number ("out"), source AS ("sas"), Destination AS ("das"), source mask ("smk"), destination mask ("dmk"), destination Tos ("dtos"), Direction: ingress, egress ("dir"), next-hop IP address ("nh"), BGP next-hop IP address ("nhb"), source vlan label ("svln"), destination vlan label ("dvln"), input source MAC address ("ismc"), output destination MAC address ("odmc"), input destination MAC address ("idmc"), output source MAC address ("osmc"), MPLS label 1 ("mpls1"), MPLS label 2 ("mpls2"), MPLS label 3 ("mpls3"), MPLS label 4 ("mpls4"), MPLS label 5 ("mpls5"), MPLS label 6 ("mpls6"), MPLS label 7 ("mpls7"), MPLS label 8 ("mpls8"), MPLS label 9 ("mpls9"), MPLS label 10 ("mpls10"), client latency ("cl"), server latency ("sl"), application latency ("al"), router IP addresss ("ra"), engine type/ID ("eng"), exid (which contains only the value 1, so this field is not considered) and the time flow received by the collector ("tr"). The lack of values for these 37 columns can be seen as a limitation of the algorithm, since their presence may generate some valuable knowledge and can reduce the number of steps initiated for cleaning and aggregating the data. For instance, the absence of the TCP flag does not guarantee that the 3-way handshake actually occurred and cannot provide additional information, such as the "push" or the "reset" flags.

The choice for the data set used to evaluate the findings from the first data set is based on the fact that it also represents an anonymized enterprise traffic, so that both files can be comparable in terms of the analysis outcome. The flows in the second data set are between 16:11:44 and 17:11:54, thus one hour and 10 seconds. The protocols used are TCP, UDP and ICMP and the traces contain the data exchange between 388 hosts. The flows describe the internal enterprise traffic registered at a medium-sized site by LBNL/ICSI Enterprise Tracing Project, which is a program supported by Berkeley Lab — Lawrence Berkeley National Laboratory[19]. This project is sponsored by Department of Homeland Security[20] and National Science Foundation[21]. The file has the form *.anon and is in tcpdump/pcap save format.

[19] https://www.lbl.gov/

[20] https://www.dhs.gov/

[21] https://www.nsf.gov/

It is confirmed that by choosing this specific data set from LBNL to validate the results obtained for the first data set, a large amount of the communication patterns discovered by means of SNA and conventional survey data do not depend on:

- The total duration of the traffic (5 minutes and 12 seconds versus one hour and 10 seconds).
- The time of the day (morning, respectively late afternoon).
- The region where the traffic was captured (Austria versus California).

3.3 Data Transformation with Ubuntu and Wireshark

As mentioned in Section 3.2, the first data set is transformed from the binary format NFCAPD into CSV by using Ubuntu, so that it could be further read and processed in RStudio. Since also the second PCAP file has to be transformed into a R readable format, it has to be chosen between converting the data into NFCAPD format and then from NFCAPD to CSV format or to use a tool for importing the PCAP file and exporting it as CSV. The second path is embraced, because it requires only one step of data transformation instead of two. The data is converted with Wireshark, since it is a wide-used network protocol analyser and provides various ways of data format and export.

The second dataset contains more attributes than the first one, so only the common attributes of the two datasets have to be kept for the purpose of a comparable analysis. The following screenshot (Figure 1) from the Wireshark environment shows how the output format of the second data set (and of any other PCAP file) can be configured.

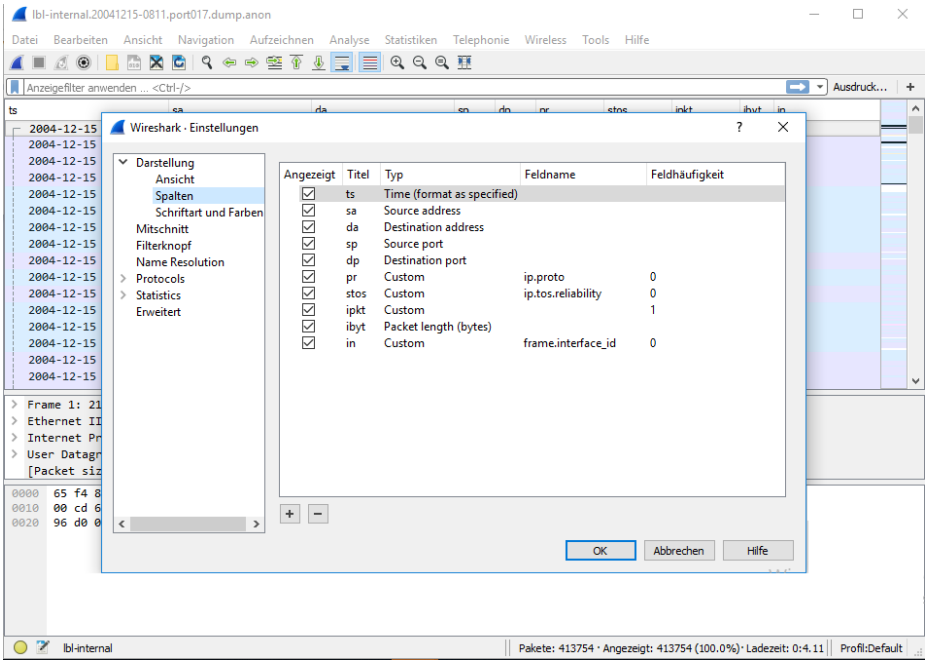

Fig. 1. Configuration of the Wireshark Output Format of the Flows

As it can be observed in Figure 1, the resulted columns headers are configured to have the same names as the predefined element tags of nfdump. Next, the time format has to be set in the format yyyy-mm-dd hh:mm:ss.123456 (Figure 2), which is the standard time format of nfdump.

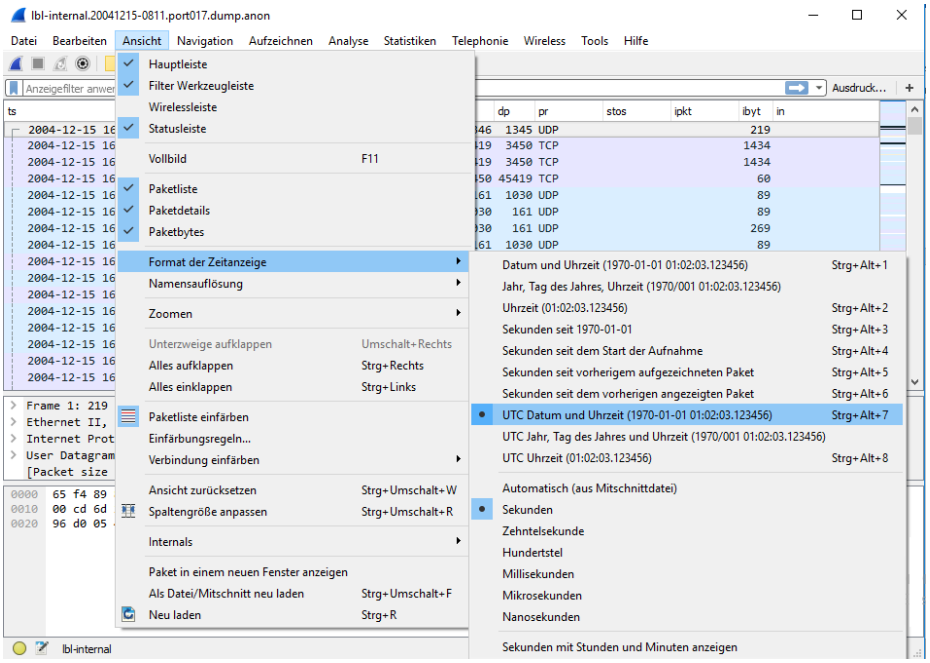

Fig. 2. Configuration of the Time Format in Wireshark

These steps preceding the export of the traces as CSV are necessary, so that the R script can transform correctly the files into data frames.

3.4 Data Pre-Processing and Data Processing

The basic flow records collected need to be processed prior to data analysis. After data understanding and its transformation into a common format for both data sets, the traces are processed with RStudio, which is an integrated development environment (IDE) for R. Among other things, it contains a console, tools for plotting and debugging and workspace management. The script contains all steps of setting up the working environment and of cleansing, wrangling, sorting, aggregation, analysis and visualization of the data.

The working environment is set by installing and loading all the necessary packages, establishing the working directory and reading the CSV data already transformed in Section 3.3. The data is mapped from the “raw” format to a data frame, with the intent of making it appropriate for the analysis process. During the data processing step, the application specific knowledge is applied to the data, by first aggregating the flow-records for the same interaction into one single record, and then integrating two unidirectional records into one edge record.

3.4.1 Combining Flow Records for the same Interaction

The source and destination end-points determine a connection, which is named a 5-tuple, since it includes five chunks of information: the protocol, the source IP address, the destination IP address, the source port and the destination port [17, p.556]. Flow records that define the same interaction are characterized by a 5-tuple. All flows consisting of identical tuples are merged to one single record, where the start time for this record is the minimum time of all the corresponding flows and the end time is the maximum time of all flows. For this purpose, the initial time dimension of the data frame is split in two (respectively into start time and end time), since this aggregation makes it possible to ascertain the time interval in which one single interaction occurs. The following screenshots out of the RStudio global environment describe the flows corresponding to the tuple (146.17.26.89, 146.16.224.49, 63466, 80, TCP) before (Figure 3) and after the aggregation (Figure 4):

	Time	SourceIP	DestIP	SourcePort	DestPort	Protocol	SourceTos	InPackets	InBytes	InInterface
	All	26.89	224.49	[...]	All	All	All	All	All	All
790	25.10.2016 11:00:13:013013	146.17.26.89	146.16.224.49	63466	80	TCP	0	18	23042	9
1295	25.10.2016 11:00:26:026026	146.17.26.89	146.16.224.49	63466	80	TCP	0	9	11521	9
2727	25.10.2016 11:00:52:052052	146.17.26.89	146.16.224.49	63466	80	TCP	0	18	23042	9
4154	25.10.2016 11:01:31:131131	146.17.26.89	146.16.224.49	63466	80	TCP	0	9	11521	9
5860	25.10.2016 11:02:10:210210	146.17.26.89	146.16.224.49	63466	80	TCP	0	0	0	9
6624	25.10.2016 11:02:23:223223	146.17.26.89	146.16.224.49	63466	80	TCP	0	9	11521	9
7912	25.10.2016 11:02:50:250250	146.17.26.89	146.16.224.49	63466	80	TCP	0	9	11521	9
9891	25.10.2016 11:03:29:329329	146.17.26.89	146.16.224.49	63466	80	TCP	0	9	11521	9
11276	25.10.2016 11:04:08:4848	146.17.26.89	146.16.224.49	63466	80	TCP	0	18	23042	9
11695	25.10.2016 11:04:20:420420	146.17.26.89	146.16.224.49	63466	80	TCP	0	9	11521	9
13018	25.10.2016 11:04:47:447447	146.17.26.89	146.16.224.49	63466	80	TCP	0	18	23042	9

Fig. 3. Tuple (146.17.26.89, 146.16.224.49, 63466, 80, TCP) before Aggregation

The aggregation assumes also the summation of all packets and bytes, which corresponds to the total transfer of the flows in one direction:

	StartTime	EndTime	SourceIP	DestIP	SourcePort	DestPort	Protocol	SourceTos	InInterface	InPackets	InBytes
	All	All	26.89	224.49	[...]	All	All	All	All	All	All
790	2016-10-25 11:00:13	2016-10-25 11:04:47	146.17.26.89	146.16.224.49	63466	80	TCP	0	9	126	161294

Fig. 4. Tuple (146.17.26.89, 146.16.224.49, 63466, 80, TCP) after Aggregation

Most of the flows corresponding to one single record have corresponding flows in the reverse direction as well (Figure 5, Figure 6), i.e., the source and destination IP addresses use the source and destination ports in a swapped way (the source in one direction is the same as the destination in the opposite direction).

	Time	SourceIP	DestIP	SourcePort	DestPort	Protocol	SourceTos	InPackets	InBytes	InInterface
	All	224.49	26.89	All	[...]	All	All	All	All	All
772	25.10.2016 11:00:13:013013	146.16.224.49	146.17.26.89	80	63466	TCP	0	10	1938	51
1298	25.10.2016 11:00:26:026026	146.16.224.49	146.17.26.89	80	63466	TCP	0	5	969	51
2728	25.10.2016 11:00:52:052052	146.16.224.49	146.17.26.89	80	63466	TCP	0	10	1938	51
4155	25.10.2016 11:01:31:131131	146.16.224.49	146.17.26.89	80	63466	TCP	0	5	969	51
5862	25.10.2016 11:02:10:210210	146.16.224.49	146.17.26.89	80	63466	TCP	0	0	0	51
6630	25.10.2016 11:02:23:223223	146.16.224.49	146.17.26.89	80	63466	TCP	0	5	969	51
7921	25.10.2016 11:02:50:250250	146.16.224.49	146.17.26.89	80	63466	TCP	0	5	969	51
9892	25.10.2016 11:03:29:329329	146.16.224.49	146.17.26.89	80	63466	TCP	0	5	969	51
11269	25.10.2016 11:04:08:4848	146.16.224.49	146.17.26.89	80	63466	TCP	0	10	1934	51
11691	25.10.2016 11:04:20:420420	146.16.224.49	146.17.26.89	80	63466	TCP	0	5	969	51
13016	25.10.2016 11:04:47:447447	146.16.224.49	146.17.26.89	80	63466	TCP	0	10	1938	51

Fig. 5. Tuple (146.16.224.49, 146.17.26.89, 80, 63466, TCP) before Aggregation

	StartTime	EndTime	SourceIP	DestIP	SourcePort	DestPort	Protocol	SourceTos	InInterface	InPackets	InBytes
	All	All	224.49	26.89	All	[...]	All	All	All	All	All
772	2016-10-25 11:00:13	2016-10-25 11:04:47	146.16.224.49	146.17.26.89	80	63466	TCP	0	51	70	13562

Fig. 6. Tuple (146.16.224.49, 146.17.26.89, 80, 63466, TCP) after Aggregation

As it can be noticed for this particular example, the time of each flow record in one direction (Figure 4) is the same as the time of the corresponding flow in the opposite direction (Figure 6).

3.4.2 Aggregating two Unidirectional Flow Records into one Record

The 5-tuple is used again to aggregate the flows aggregated in the previous step into an edge record, so that both the well-known services, the P2P and the illegitimate traffic can be identified. An edge record defines a client-server interaction, respectively a P2P one. For the protocols TCP and UDP two single records are aggregated into one record if the flows use both the source/destination IPs and the source/destination ports in a reversed manner. For ICMP, the records are aggregated if they are between the same IP addresses, without considering the ports, since ICMP does not have a port abstraction. For instance, the single flow record in Figure 4 (146.17.26.89, 146.16.224.49, 63466, 80, TCP) is combined with the one in Figure 6 (146.16.224.49, 146.17.26.89, 80, 63466, TCP) into one edge-record (Figure 7):

StartTime	EndTime	SourceIP	DestIP	SourcePort	DestPort	Protocol	OutTos	OutInterface	OutPackets	OutBytes	OutFlows	OutAvgPacketSize	SourceTos	InInterface	InPackets	InBytes	InFlows	InAvgPacketSize
All	All	26.89	224.49	[...]	All	All	All	All	All	All	All	All	All	All	All	All	All	All
2016-10-25 11:00:13	2016-10-25 11:04:47	146.17.26.89	146.16.224.49	63466	80	TCP	0	9	126	161294	11	1	0	51	70	13562	11	NaN

Fig. 7. Edge Record of Tuples (146.16.224.49, 146.17.26.89, 80, 63466, TCP) and (146.16.224.49, 146.17.26.89, 80, 63466, TCP)

Since the start and the end time of two single paired records can be also different, the time interval of the edge record is defined as follows:

– The start time is the minimum start time of the two single records.

– The end time is the maximum end time of the two single records.

By using this approach, also the time-delayed response-flows corresponding to a specific 5-tuple are encapsulated into the aggregation. Concurrently with the aggregation into an edge-record, the columns corresponding to the bytes, packets and type of service, interface are split in two, as a result of bidirectionality. Now, also the attributes of the reversed direction of the single record are mapped into the data frame. The columns corresponding to the average packet size, which is defined as the division of bytes per packet, are used in Section 3.7 for the traffic classification into P2P and not P2P.

The single records with no response found in the opposite direction are removed, since their retention as unidirectional flows together with the bidirectional flows can produce bias during the further analysis. A possible cause for the missing corresponding record in the opposite direction is that the response lays in another time interval, which is beyond the timespan of the data set.

The result of combining the flows together is an edge-record and the data is represented as a graph that has nodes as unique hosts and edges that define the communication between these hosts.

In order to simplify the further steps of the analysis, all destination ports are assigned values lower than 1024, by assuming a client/server interaction. This approach leads to a more efficient computational effort when used to identify the servers, since the script searches only in one dimension of the data frame, thus in the destination port field. As noted by the authors in [20, p.88], who use the historical community of interest of a host to anticipate both forthcoming behavior of the IP network and the behavior of other members of this community, the flows having a client port number (source port) above 1024 and a server port number (destination port) below 1024 are in the majority of the cases correct, since it is consistent with the normal usage of reserved ports. On the other side, the reverse approach (server port above 1024 and client port below 1024) is likely to be incorrect, as assessed by [20, p.88] with an experimental error of 2,187%. Since client/server directionality is not considered, due to the existence of P2P connections within the network, this role assignment error does not affect the results of the further analysis.

3.4.3 Remove illegitimate Traffic

Since only the “proper” traffic within the network presents interest for the analysis, all graph edges representing the traffic suspected to be unwanted (worms or scans) are removed. Following rules for this cleansing step are used, according to the protocols used:

– Since a valid application layer data transport requires at least three packets to open, transmit and close a TCP connection [20, p.6], the edge records using TCP as protocol and having less than 3 packets in each direction are removed. This step eliminates ca. 18% of all edges in case of the first data set

and 10% in case of the second data set, which signifies that some of the flows do not fulfill an application-level data transfer.

– UDP can be used as a request/response interaction (DNS, RPC) or in streaming application (long lived UDP flow [21, p.1]). As a result, it is expected that a “useful” UDP flow is correlated with the transfer of at least two packets, either in one direction or in the opposite one [20, p.6]. For that reason, all edge records where the sum of the packets sent and the packets received is smaller than 2 are removed. Both first data set and second data set contain only legitimate UDP flows.
– There is no cleaning applied on ICMP traffic, since one ICMP datagram is a legitimate use of ICMP [20, p.6].

Applying these cleaning methods on the data leads to a retention of 82% of the total edge records for the first data set and of 90% of the total edge records for the second data set.

3.5 Identification of Anomalous Traffic

The identification of anomalous traffic follows the step of data preprocessing and processing. Following illustration (Figure 22) addresses the possible problems that could occur within the network traffic and their classification into five categories:

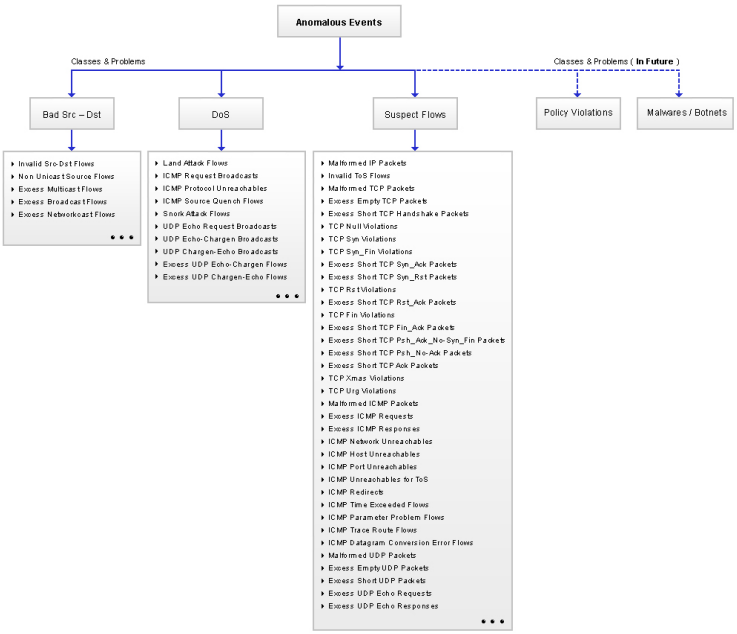

Fig. 8. Anomalous Events [22]

Out of the problems enumerated in Figure 8, only some of them are profiled, since the data sets containing the monitored traffic do not include enough information available at the perimeter gateways in order that all of these issues can be identified. For instance, the identification of "TCP Syn Violations" requires the knowledge about the TCP flag, which is missing in the sample data. Another example is the identification of "Non Unicast Source Flows", which requires the precondition to know if the IP address is Multicast, Broadcast or Unicast. This field is also missing in the network data. Firstly, the suspect flows are identified, by applying multiple heuristics on the source and destination ports. The type of service (ToS) field specifies the datagram priority and is used to request a particular handling such as high throughput or low latency. The ToS values can be specified in the interval (0, 255), although only following values are valid: 0, 32, 40, 48, 56, 64, 72, 80, 88, 96, 104, 112, 120, 128, 136, 144, 152, 160, 184, 192 and 224 [44]. If a flow contains other ToS values than the ones specified before, then this flow is marked as invalid ("invalid_ToS").

Following heuristics for identifying the suspect flows and denial of service attacks are based on the profiling methods described by [22]. The ICMP flows meaning an unreachable network are identified by filtering out the destination ports 768, 774, 777 or 779 and the protocol ICMP (marked as "ICMP_network_unreachable") [22]. ICMP flows defining unreachable hosts have the destination ports 769, 773, 775, 776, 778, 780 and 781 (marked as "ICMP_host_unreachable") [22]. ICMP flows standing for unreachable ports have the destination port 771 (marked as "ICMP_port_unreachable") [22]. ICMP flows unreachable for ToS have the destination port 779 or 780 (marked as "ICMP_ToS_unreachable"). Only the field of the destination port is used to identify the flows enumerated above, because this field contains values below 1024, as determined in the pre-processing phase. ICMP redirect flows have either the source port or the destination port in 1280, 1281, 1282 or 1283 (marked as "ICMP_Redirects") [22]. ICMP Time Exceed flows signifies traceroute attempts or a failed reassembly of datagram fragments and have either the source port or the destination port 2816 or 2817 (marked as "ICMP_Time_Exceeds_Flows") [22]. ICMP Parameter Problem Flows signifies a local implementation error and have either the source port or the destination port 3072, 3073 or 3074 (marked as "ICMP_Parameter_Problem") [22]. Flows having either the source port or the destination port 7680 indicate a traceroute attempt (marked as "ICMP_Trace_Route") [22]. Flows having either the source port or the destination port 7936 specify an error in converting the datagram (marked as "ICMP_Datagram_Conversion_Error") [22]. Flows having the average packet size below 20 transfer malformed IP packets (marked as "malformed_packets") [22]. The latter approach is restrained to TCP, UDP and ICMP traffic. All TCP flows having the average packet size below 40 transport malformed TCP packets (marked as "malformed_TCP_packets") [22]. Analogously, all ICMP flows having the average packet size below 28 transport malformed ICMP packets (marked as "malformed_ICMP_packets") and all UDP flows having the average packet size below 28 have malformed UDP packets (marked as "malformed_UDP_packets") [22]. The UDP flows having exactly 28 bytes as average

packet size are marked as “excess_empty_UDP_packets” [22]. All the 5-tuple flows, which fulfill the heuristics above, are mapped in the column “SuspectFlow” of the data frame with their corresponding label.

Secondly, the flows that are suspected for denial of service (DoS) attacks are mapped in a separate column. These attacks usually generate deviations in the traffic volume that are detectable at flow level [19, p.348]. Further, they can be categorized into land attacks and snork attacks. Flows having the same IPs indicate that the target machine replicates to itself repeatedly (marked as “land_attack”). Flows having either the source or the destination port 135 and the corresponding port in the opposite direction 7, 19 or 135 specify a denial of service attack [22]. They are labeled as “snork_attack”.

Thirdly, possible worms and/or Trojan horses are identified, based on the ports and on the number of bytes or packets exchanged. The approach of identifying malicious activity is based on the heuristics proposed by [23, p.1082]. Trojan horses are designed primarily to allow hackers having access to system files, which results in damaged files, stolen passwords or changed file settings [24, p.22]. Worms are programs originating on a computer and then searching for other computers in a local area network or through Internet connection with the intent of replicating themselves [24, p.23]. According to [23, p.1081], Trojan horses and worms with typical features can be identified using the distinctive fields containing the port numbers and the bytes transfer. The most common worms/Trojans are profiled in the column “Worms_Trojans” of the main data frame. As stated in [23, p.1082], ‘‘Wincrash_v2” is defined by the source or destination port 2583 and the TCP protocol. The worm ‘‘Shockwave Killer” uses the source or destination port 2048, the ICMP protocol and 92 transfer bytes either in one direction or in the reversed one [23, p.1082]. Code Red Worm has the source or destination port 80, the TCP protocol, a number of 3 packets and 144 bytes [23, p.1082]. Worm.Opasoft and W32.Opaserv.Worm have the destination port 137, the UDP protocol and 78 transfer bytes. Worm.NetKiller2003, Worm.Sqlp1434, W32.Slammer and W32.SQLExp.Worm have the source or destination port 1434, the protocol type UDP and 404 transfer bytes [23, p.1082]. Worm.Blaster and W32.Blaster.Worm use the source or the destination port 135, the TCP protocol and 48 transfer bytes [23, p.1082]. Worm.KillMsBlast, W32.Nachi.worm and W32.Welchia.Worm have the source or destination port 2048, the protocol type ICMP and 92 transfer bytes [23, p.1082]. Worm.Sasser and W32.Sasser have the source or destination port 445, the TCP protocol and 48 transfer bytes [23, p.1082]. W32.Witty.Worm uses the source or destination port 4000 and the UDP protocol [23, p.1082]. For all the heuristics above, the transfer bytes and packets are considered in both directions (i.e., both corresponding to the source and destination host) when profiling the malicious activity.

The results of the heuristics above – suspect flows, denial of service attacks and worms/trojans are introduced in a separate column for each of these categories, in

which the identified anomaly is documented. The 5-tuples corresponding to the anomalous flows identified in this step are inserted in a separate table and are visible in the HTML report, as well.

3.6 Identification of Common Services

In this phase of the analysis, the common services such as web servers, client web, email, domain name system (DNS), virtual private networks (VPN) and remote services are profiled, by using the methodology proposed by [4]. The sample data in [4] was collected and profiled with SiLK[22], which simplifies the security analysis of large networks, by using a set of traffic analysis tools developed by the CERT Network Situational Awareness Team (CERT NetSA). Additionally to the approach proposed by [4], the current methodology extends the ports list used for the profiling and also allows the visibility of the results on host and flow level, which enables finding similarities between the flows. It is also visible if a host, to which a specific type of service is assigned, also participates in connections with hosts having different attributes.

Following columns result out of the profiling of common services: web servers, web clients, mail servers, mail clients, DNS servers, DNS clients, recursive DNS, VPN, remote servers and remote clients. Concurrently with the identification of these services, also the top protocols, the top destination ports and the top source ports are determined and can be visualized in form of pie charts. These charts are visible in the HTML report, as well.

3.6.1 Web Servers

Most of the business concerns are led by web traffic and email, where web servers represent a large amount of web traffic. The first step of the profiling includes looking for network traffic running on the destination ports 80, 8080, 443, 81, 82 and 8090 and using the TCP protocol. The approach in [4, p.26] takes into consideration only the ports 80, 8080 and 443, but web servers can listen also to the alternate ports 81, 82 and 8090. Only the IP addresses that transfer more than 1% of the total bytes are profiled, since these are the most hectic web servers, according to [4, p.14]. The top 7 web servers can be visualized in the HTML report but all common services including web servers are mapped on flow level in the data frame containing the whole traffic, which is referred further on as the main data frame.

The list of the potential web servers needs to be validated by using a web browser, since some flows containing these web servers do not necessarily represent web traffic. If possible, the validation of the findings should be done outside the network, since the server may also receive other traffic types from the internal network. If there

[22] https://tools.netsa.cert.org/silk/

are a few external addresses connecting to one of the web servers, then this web server might be isolated for a few addresses or a point-to-point SSL VPN [4, p.16]. On the other hand, the presence of many external addresses indicates that it is an actual web server.

This algorithm can generate some false positives, which can be examined by looking at all flows in the main data frame labeled as web servers:

- Connections having a long-duration and a high transfer on port 443 can represent SSL VPN traffic [4, p.17].
- Web servers with many client connections can be gateways [4, p.17].
- It might be the case that a server uses more than one IP address or that many websites use one single IP address; in order to find this out, the IP addresses can be concluded to DNS names with DNS lookup, respectively DNS forward lookup [4, p.17].
- Some servers use the ports 1935, 1755 and 554 for streaming media. If a previously identified web server uses also one of these ports, then this server provides a streaming media service [4, p.17].

3.6.2 Web Clients

The approach for identifying the web clients is opposite to the one used for detecting web servers, thus flows going to web ports on an external host are considered. Namely, the web clients are identified by profiling the source IP addresses running on the destination ports 80, 8080, 443, 81, 82 or 8090 and using the TCP protocol. Only the hosts using at least 1% of the total byte transfer of the traffic are considered [4, p.20]. The web clients list can be further categorized into proxy servers, NAT gateways and directly connected workstations, by looking at both inbound and outbound traffic [4, p.21]. A client that hosts many services can also be a VPN gateway.

False positives can arise due to the following patterns:

- If there is only one client web traffic to an IP address, then this address should not be considered as a web client, since servers receive sometimes for their updates a minimal amount of traffic from some external addresses [4, p.23].
- To simplify application handling but also to avoid firewall policies, any service can be configured to use the ports designated by default for web traffic [4, p.23].

3.6.3 Email Servers and Email Clients

The email servers are profiled by filtering out the destination ports 25, 465, 110, 995, 143 or 993 and the protocol TCP. The destination IP addresses corresponding to these

entries are the web servers of the network. The email clients are the source IP addresses running on the destination ports 80, 8080, 443, 81, 82 or 8090 and using the TCP protocol.

Email servers can be validated by using nslookup or the mail exchanger record of the domain [4, p.25]. Email clients can be certified by searching the web for a specific client address to see if it appears in any email headers or by looking for patterns of expected client traffic into the sample data [4, p.26].

Inconsistencies in profiling email service can arise due to the following issues:

- Email servers can use also other services, for instance web. Thus, an email server can be profiled as a web server, as well [4, p.26].
- Also a client can run an email server, since email services do not necessarily run on dedicated servers. If this is the case, the policies concerning clients running on these services and the firewall configurations need to be checked [4, p.26].
- Desktop clients which send SMTP messages should be investigated, i.e., all clients that send messages straight to the internet [4, p.26].

3.6.4 Domain Name System

The Domain Name System (DNS) enables the translation between domain names and IPs [4, p.28]. The DNS servers are identified by filtering out the hosts which accept connections on port 53 and the clients by profiling the hosts talking to services using the port 53 [4, p.29]. DNS uses both TCP and UDP for its transactions but since TCP is used only in case of zone transfers, which rarely occur across the borders of the perimeter, only the flows using UDP are analyzed [4, p.28]. The traffic volume is measured by means of packets instead of bytes, since each DNS request is encapsulated in a single UDP packet [4, p.28]. Thus, in case of both DNS servers and DNS clients, only the IP addresses which transfer at least 1% of the total packets are examined [4, p.29].

If an IP address is in both lists of DNS clients and DNS servers, then this IP is a recursive name server, since it both receives and sends requests (Fig.9).

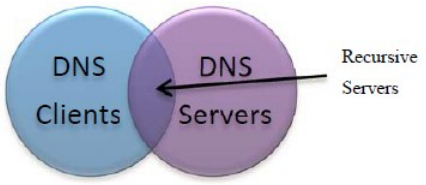

Fig. 9. Recursive DNS [4, p.30]

The DNS servers can be validated by concluding the IPs to a domain name. Some anomalies might appear when analysing DNS flows:

- Server-to-server requests on port 53 are the consequence of using an older implementation of DNS [4, p.31].
- If there are DNS responses from any host except the ones belonging to the network's DNS servers, this could signify a suspicious activity or a violation of the policy [4, p.32].

3.6.5 Virtual private Networks

Virtual private networks (VPN) are a group logical nodes and virtual paths [26, p.1] and can be seen as tunnels over the public networks, allowing the connection of the users to the main office network [4, p.33]. The setup phase enables the initialization of the VPN, even if the target IP is behind a NAT machine, and the tunneling phase has usually no ports linked to the protocols used, such as ESP and GRE, making VPN different from the other NetFlow traffic [4, p.33].

The VPN gateways are profiled by looking at the IP addresses that use the protocols ESP, GRE and AH and transfer at least 1% of the total bytes and they can be validated with nslookup or by looking at traffic patterns [4, p.34]. If an asset performs short negotiations over the UDP protocol and has a long VPN session over ESP, GRE or AH, which is refreshed every hour, then this asset is likely to be a VPN gateway [4, p.36].

Also in case of VPN traffic, some inconsistencies need to be taken into consideration:

- VPN can run also on non-standard ports, which have to be manually configured so that they can have access through the firewall [4, p.36].
- Distinct timeouts can be visible across VPN, depending on the vendor's choice. Independent of the timeout length, the patterns within the flows remain the same [4, p.36].

3.6.6 Remote Servers and Remote Clients

A characteristic of remote services is that the data contains small packets on the command channel and large ones on the data channel. Three types of remote services are profiled: FTP (port 21), SSH (port 22) and Telnet (port 23), by individually outlining the destination IP addresses running on these destination ports (servers) and also the source IP address running on these ports (clients). Only the assets that contribute to more than 1% of the remote traffic are examined [4, p.38]..

For validating the findings, nslookup can be used. FTP sessions in passive mode can be validated if the traffic preceding a high-port-to-high-port connection contains

an edge record running on port 21 [4, p.41]. FTP sessions in active mode can be validated if the traffic pattern has a high port as source port and port 20 as destination port [4, p.42].

Possible anomalies may arise due to the following issues:

- Some FTP servers are FTP clients, as well [4, p.42].
- Timeouts (no server response), due to the configuration of the firewalls [4, p.42].
- Control channel without data channel, due to either the active use of keep-alive packets (which are null) or to brute force attacks on the FTP servers [4, p.42].

3.7 Identification of P2P Traffic

The profiling in Section 3.6 assumes a pure client-server architecture of the network where all the shared resources exist exclusively on the server. Thus, the clients and servers are profiled separately. If a security issue is suspected around one of the shared resources, then the responsible server needs to be investigated closer. This can be accomplished in a straightforward manner, by analyzing the corresponding server lists, which are created as a result of the methodology described in Section 3.6.

In addition, P2P traffic can be existent within the data sets, particularly because this type of traffic continues to grow steadily and occurs very frequently inside a closely working group of people in an enterprise network. Compared to the traditional client-server model, peers are both suppliers and consumers of resources, meaning they show an increased exposure to remote attacks. Therefore, P2P networks are insecure by nature, because there is no centralized server steering the access to the shared resources, but these resources are located on the local machines. In a P2P network, each user is responsible for providing controlling access to the resources existing on the own computer. The network cannot check the login names to determine if the user is authorized to access the share. P2P network traffic works very different so it needs to be profiled differently. In addition, due to its security implications, it is emphasized more than the traffic corresponding to the common services in course of the further analysis.

The identification of P2P flows is important for adequate network planning and design, security assurance, network management and understanding of network behavior [27, p.296]. Opposed to the first-generation P2P networks, which had well designated port numbers, the contemporary P2P applications can cover their presence by using arbitrary ports and custom-designed protocols, so that they can bypass filtering firewalls and legal issues. Thus, an accurate profiling of P2P traffic requires the inspection of packet payload, as well. Nevertheless, the user payload is most of the times inaccessible, due to legal, technical or logistic matters. This hinders a trustworthy approximation of the expansion and dynamics of P2P traffic. [5] developed a methodology for identifying P2P flows based on connection patterns of P2P networks, without

considering packet payload. Since [5] validated the findings by using a payload technique and achieved by that an accuracy of 99% percent of the non-payload method, a similar algorithm is used to identify the P2P connections within the two data sets. Related with the identification of P2P traffic suggested in [5], the flows are classified in two categories: P2P and non-P2P. Additionally, the methodology proposed by [5] is extended, by taking into consideration the whole network traffic instead of a limited portion of it when applying the methods for identifying the false positives. Also [28] profiles the P2P traffic within a Tier 1 provider backbone, thus only the fixed port numbers are taken into consideration, which leads to an incomplete profiling of the P2P traffic.

In order to minimize the false positives in identifying the P2P traffic the connection characteristics of various protocols and services identified in Section 3.6 (for instance web traffic, mail and DNS) are evaluated and filtered. Even if some of these patterns are not exclusive for P2P, inspecting the flow history or possible scans can help removing the false positives and disclose unique characteristics. The data frame obtained after processing the data and mapping the common services represents the flow table used. This data frame contains all the various flow characteristics required for the identification of P2P traffic.

In the first step, all source IP and destination IP pairs, which use both protocols TCP and UDP for their interactions, are identified, since most of the P2P protocols use concurrently UDP for control traffic and queries and TCP for actual data transfer. In addition, other services such as DNS, NETBIOS, NTP, ISAKMP, IRC, gaming and streaming media use TCP and UDP simultaneously. Following ports are assigned to the services mentioned above: NETBIOS - 135, 137, 139 and 445, DNS – 53, NTP – 123, ISAKMP – 500, streaming media – 554, 1755, 5000, 5001, 6970, 7070, 6970, IRC – 7000, 7514, 6667, gaming – 6112, 6868, 6899 [5, p.124]. Therefore, a (IP, IP) pair is categorized as P2P if it uses simultaneously TCP and UDP and does not use the ports assigned to the services mentioned above. All flows containing one of the IPs in these pairs are flagged in the main data frame as P2P.

In the second step, the P2P flows are identified by analyzing the connections patterns of the (IP, port) pairs, respectively the number of IPs and the numbers of ports they connect to. An (IP, port) pair is an affiliation of a source or destination IP and a source or destination port.

Since distributed P2P networks appeared, each client has a starting host cache, which contains the IP addresses of other hosts that enable the initial connection of the P2P network [5, p.125]. These hosts can be peers, servers or hosts handling routing and query propagation (superpeers). As reported by [16, p.3], a peer sends a negotiation request to another peer having the same resource. If this request is acknowledged, another request for data transfer is sent. According to [5, p.125], when a host connects to a superpeer, either over TCP or over UDP, then this host informs this superpeer of its IP and port to which it can be reached from other peers. This (IP, port) pair becomes the host's ID, which is used by other peers to connect to it. The superpeer sends the (IP, port) information to the rest of the P2P network. In case that many

peers connect to a single host, each one of them chooses a temporary source port to connect to the listening port announced by this host. Thus, the number of distinct IP addresses linked to this particular host is the same as the number of distinct ports connected to it. Figure 10 explains this process.

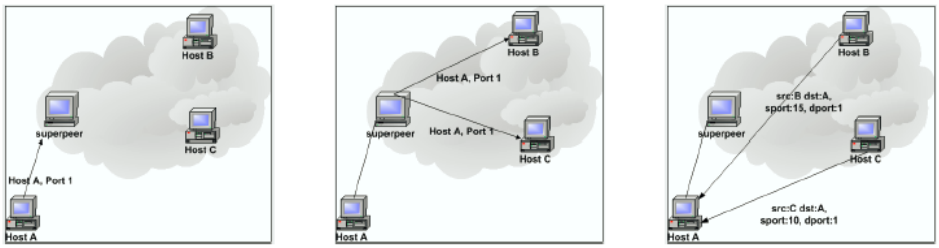

Fig. 10. Connection of a P2P Host to the P2P Network [5, p.125]

On the contrary, in case of HTTP for instance, a host connecting to the web will start at least one simultaneous connection to download items at the same time.

Therefore, all (IP, port) pairs which connect to the same number of IPs and ports are labeled as P2P and all flows in the main data frame which contain one of these pairs are labeled as P2P.

Since the backbone links show a wide diversity of host types and flows, some of the identified P2P flows can be nonP2P. Thus, it is anticipated that the two procedures outlined before return false positives, which are most prevalent in pairs having a few connections and in pairs having one connection per port such as mail, DNS and gaming. Following approaches restrain the amount of false positives, by addressing the port history and reviewing the connections of all (IP, port) pairs.

Firstly, the flows having the connection patterns of DNS are analyzed, since only the true P2P flows are taken into consideration and not the DNS request of a host. Even if DNS was already profiled in Section 3.6.4, the methodology used in this section is not restrained only for DNS but it is applied for all (IP, port) pairs which use a port number below 501. This approach minimizes the amount of false positives by removing also the frequently used ports from the analysis. As a result, all flows that have the same source and destination port and a port number below 501 are flagged as nonP2P [5, p.126].

Secondly, the flows containing gaming and malware (worm and port space scans) traffic are flagged as nonP2P. Additionally to the ports addressed in Section 3.5, which are used for profiling the worms and the Trojan horses, the ports list is extended with some additional ports. Online gaming and malware activity distinguish themselves by having many connections to distinct IPs or ports and transferring a large amount of bytes/packets or packets having the same size [5, p.126]. If all flows that contain a particular (IP, port) pair have the same average packet size (which is obtained by dividing the number of bytes by the number of packets), it is most likely

that this flows indicate a malicious activity. Consequently, according to [5, p.127], an (IP, port) pair is flagged as nonP2P if:

- It has less than three mean packet sizes in all flows in which it appears.
- It does not use any of the ports used classically by P2P traffic (4661, 4662, 4665, 1214, 6346, 6347, 412, 411, 41170, 6881, 6882, 6883, 6884, 6885, 6886, 6887, 6888, 6889, 6699, 6257 or 2234).
- It either has connections to more than five IP addresses or the port is one of the ports used typically by malware (3127, 3128, 1433, 1434, 3531, 1080, 10080, 17300, 6129, 27015, 27016, 901, 2745) or the port within this pair is lower than 501.

All flows containing the pairs identified as nonP2P in this step are also marked as nonP2P in the main data frame.

Thirdly, vertical scans are identified, by calculating the number of times each IP appears in a (IP, port) pair and the number of other IPs to which this IP is connected. According to [15, p.298], a vertical scanner is expected to reach more ports on a target device than a certain client does. All pairs for which the *number of IP in pair / number of IPs connected to IP* is above 15 are marked as nonP2P. Respectively, all IP addresses that appear frequently in pairs and concurrently target a few IP addresses. All the flows in the main data frame containing one of these IPs are marked as nonP2P. The value 15 was chosen as a threshold since the experiments for both data sets having different values of the threshold show that for values below 15 also flows correctly flagged as web or DNS in the previous steps are marked as scans.

In the fourth and last step the port history is examined by flagging as nonP2P all (IP, port) pairs where the ports connected to them are below 1024 and the pair appears in less than 10 flows. Only ports below 1024 are chosen, since they are registered ports that are used mainly for standard services such as mail, DNS or HTTP. It is not probable that a P2P client connects only to well-known services, since the clients usually use a random port number for receiving the connections. For this to occur, many P2P users must change their client's listening port to at least one of the registered ports. As a result, all pairs for which all ports are registered ports and the pair appears in more than 10 flows are marked as P2P [5, p.127].

After performing these steps, the flows identified as P2P are analyzed against the flows flagged as non-P2P and only the flows that are not email, DNS, scans, port history or gaming/malware are kept as P2P.

The analysis above can be described by the algorithm in Figure 11, which is implemented in R and is inspired by the algorithm proposed by [5]. One difference between the two algorithms is that the algorithm in [5] assumes that an (IP, port) pair already classified as P2P is a false positive only when the DNS or mail heuristics for this pair are true. Additionally, the algorithm in this thesis validates an (IP, port) pair already classified as P2P also against the port history, malware and scan. Another difference between the two algorithms is that the algorithm in this thesis does not include a list containing the rejected (IP, port) pairs, which is used for finding the false positives.

The P2P algorithm identified 285 P2P flows out of 1886 total flows in the first data set (aprox. 15% P2P traffic) and 47 P2P flows out of 2319 total flows in the second data set (aprox. 2% P2P traffic).

```
algorithm P2P identification is
define DF as main data frame
define IP1 as sourceIP
define IP2 as destIP
define IP as sourceIP or destIP
define port as sourcePort or destPort
define P2PTCPUDP_DF as empty dataframe
define P2PIPPORT_DF as empty dataframe
define P2PIPPORT10_DF as empty dataframe
define P2PPORTS as dataframe containing the P2P ports
define MalwarePorts as dataframe containing the Malware ports
define mailPorts as dataframe containing the mail ports
define portHistory as dataframe containing the (IP, port) pairs where for each pair
      all connected ports are low ports and the pair appears in less than 10 flows
define count_ports as number of distinct ports connected to an (IP, port) pair
define count_ip as number of distinct IPs connected to an (IP, port) pair
define countIPinPair as number of distinct IPs within an (IP, port) pair
define averagePacketSize as bytes/packets for an (IP, port) pair
insert in DF new column P2PTCPUDP
insert in DF new column P2PIPPORT
insert in DF new column P2PIPPORT10
insert in DF new column DNS
insert in DF new column MAIL
insert in DF new column NONP2P_GamingMalware
insert in DF new column POSSIBLESCAN
insert in DF new column PORTHISTORY
insert in DF new column P2PNONP2P
for each (IP1, IP2) pair in DF do
    if protocol  = TCP AND protocol  = UDP then
        insert (IP1,IP2) in P2PTCPUDP_DF
for each (IP, port) pair in DF do
    if count_ports  - count_ip  <= 1 OR count_ip  - count_ports  <= 1 then
        insert (IP, port) in P2PIPPORT_DF
    if (count_ports  - count_ip  < 10 OR count_ip - count_ports <10)  AND port in P2PPORTS then
        insert (IP, port) in P2PIPPORT10_DF
for each row in DF do
    if (IP,port) in P2PTCPUDP_DF then
        insert "P2P" in P2PTCPUDP
    if (IP,port) in P2PIPPORT_DF then
        insert "P2P" in P2PIPPORT
    if (IP,port) in P2PIPPORT10_DF then
        insert "P2P" in P2PIPPORT10
    if (IP,port) in portHistory then
        insert "NONP2P" in PORTHISTORY
    if averagePacketSize <3 AND (IP, port) not in P2PPORTS AND
    (count_IP>5 or port in (IP,port) <501 or port in (IP,port) in MalwarePorts) then
        insert "NONP2P" in NONP2P_GamingMalware
    if countIPinPair/countIP >=15 then
        insert "NONP2P" in POSSIBLESCAN
    if SourcePort = DestPort AND SourcePort < 501 then
        insert "DNS" in DNS
    if SourcePort in mailPorts or DestPort in mailPorts then
        insert "Mail" in MAIL
for each row in DF do
    if MAIL="Mail" OR DNS="DNS" OR POSSIBLESCAN="NONP2P" OR NONP2P_GamingMalware="NONP2P" OR
    PORTHISTORY="NONP2P" then
        insert "NONP2P" in P2PNONP2P
    elseif P2PTCPUDP ="P2P" OR P2PIPPORT ="P2P" OR P2PIPPORT10 ="P2P" then
        insert "P2P" in P2PNONP2P
    elseif is.null(PORTHISTORY) then
        insert "P2P" in P2PNONP2P
```

Fig. 11. Algorithm for the Identification of P2P Flows

3.8 Graphical Representation of the known Communication Patterns

The communication patterns identified in the previous steps can be represented in a visual manner by using graphlets. The dark gray elements in Fig. 12 – Fig. 21 represent the particular pattern.

The web server pattern (Figure 12) is characterized by connections coming from the ports 443, 80, 81, 82, 8080 and 8090.

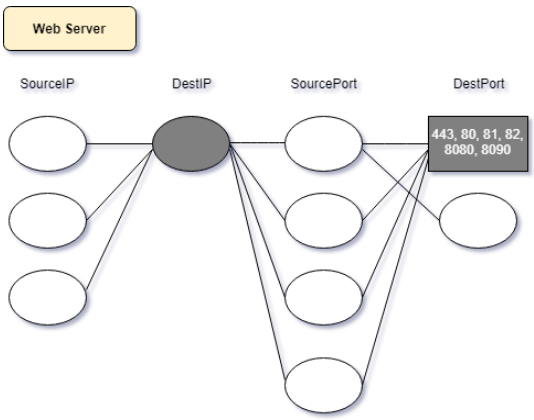

Fig. 12. Web Server Pattern

The client web pattern (Figure 13) is similar with the web server pattern but it describes the opposite process. The pattern is characterized by connections going to the ports 443, 80, 81, 82, 8080, 8090 on an external host.

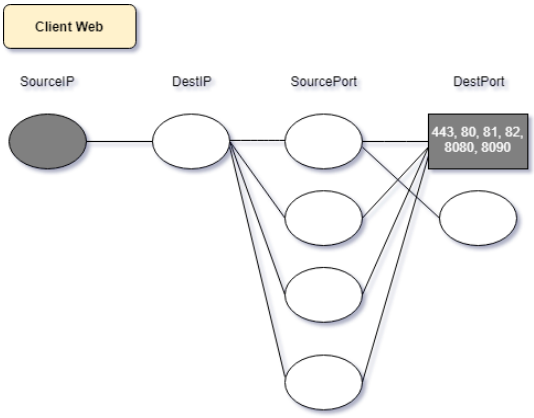

Fig. 13. Client Web Pattern

Hosts accepting traffic on port 53 characterize the DNS server pattern (Figure 14).

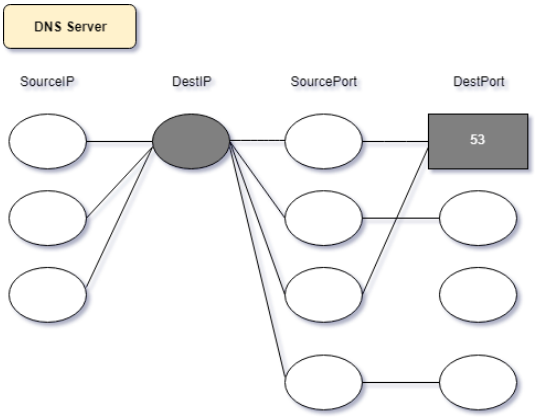

Fig. 14. DNS Server Pattern

Hosts communicating with various services on port 53 define the DNS client pattern (Figure 15).

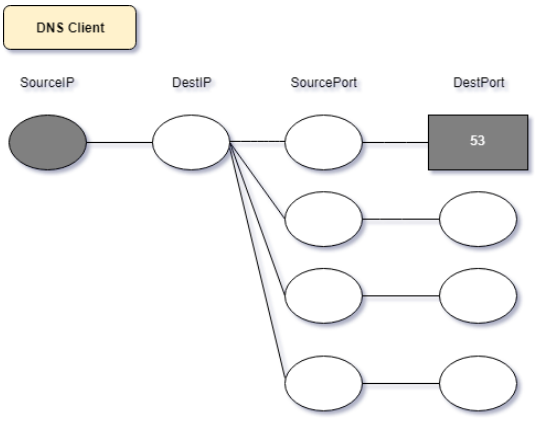

Fig. 15. DNS Client Pattern

If an IP address belongs to both lists of DNS servers and DNS clients, then this IP is a recursive name server (Figure 16) and it acts both as a client and as a server.

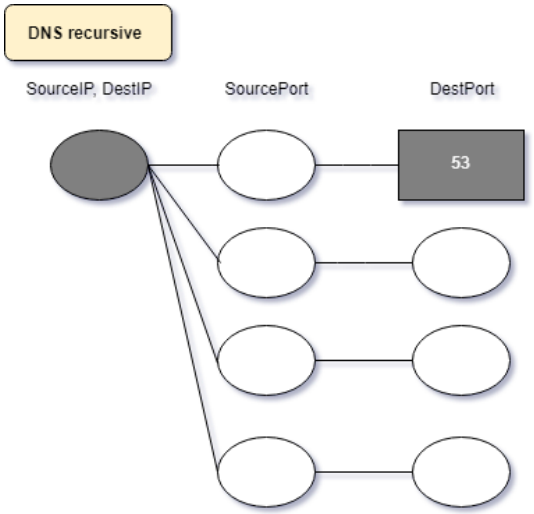

Fig. 16. DNS Recursive Pattern

The remote file server pattern is characterized by outbound flows from the ports 21, 22 or 23 (Figure 17).

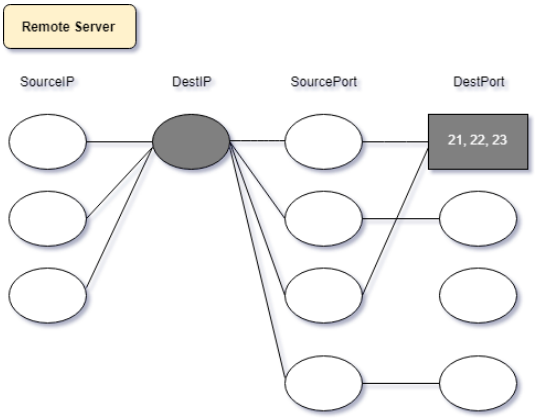

Fig. 17. Remote Server Pattern

The remote file client pattern is similar to the pattern in Figure 17, but it is defined by inbound connections to the destination ports 21, 22 or 23 (Figure 18).

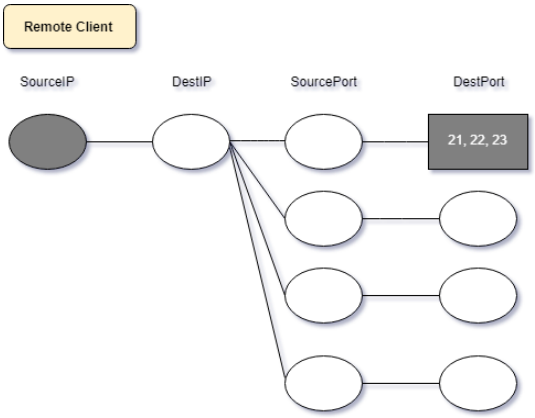

Fig. 18. Remote Client Pattern

The mail server pattern (Figure 19) is defined by traffic sent from the ports 25, 465, 110, 995, 143, 993 on an external host.

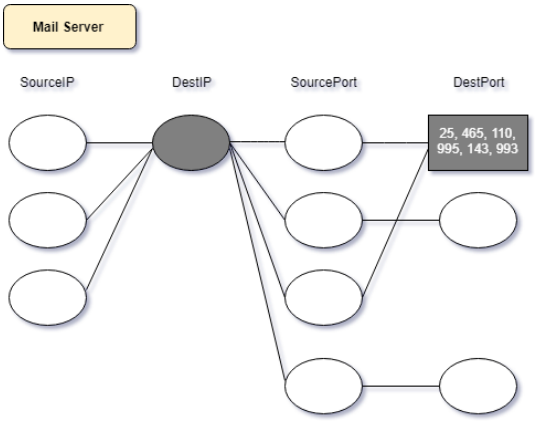

Fig. 19. Mail Server Pattern

Clients collecting messages from the ports or encrypted ports 25, 465, 110, 995, 143, 993 define the mail client pattern (Figure 20), thus the outbound connections from these ports characterize this pattern.

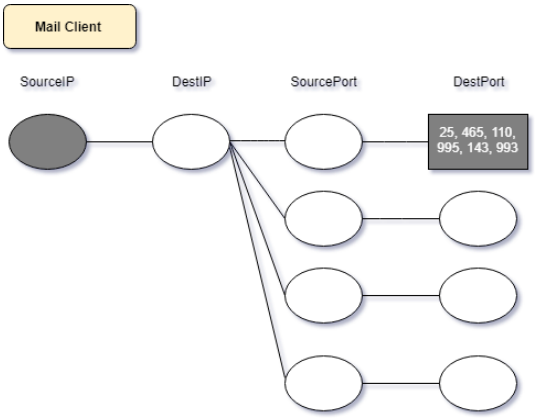

Fig. 20. Mail Client Pattern

The VPN pattern (Figure 21) occurs mainly between a remote office and the main office [39, p.34] and is characterized by the assets using the protocols ESP, GRE and AH.

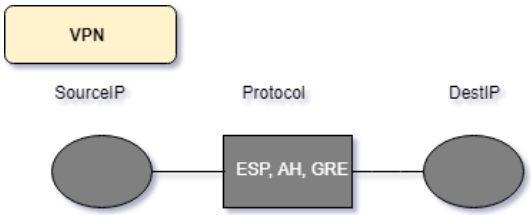

Fig. 21. VPN Pattern

According to the P2P algorithm, a P2P connection is defined by all (IP, port) pairs that connect to the same number of IPs and ports. The pattern illustrated in Figure 22 describes the same number of destination IPs and destination ports affiliated to the pair (SourceIP, SourcePort).

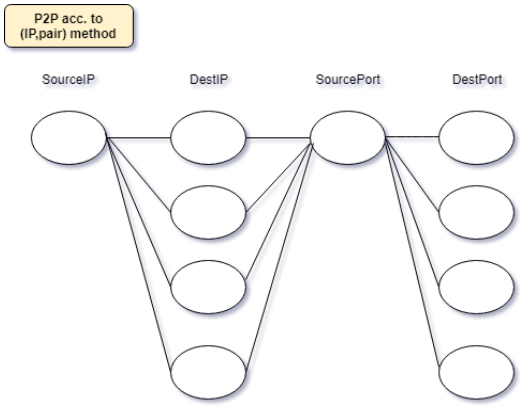

Fig. 22. P2P Pattern according to (IP, pair) Method

The presence of scans supports the evaluation of false positives within the P2P algorithm (Figure 23). Hosts connecting to a few IPs and appearing in a large number of (IP, port) pairs define scans.

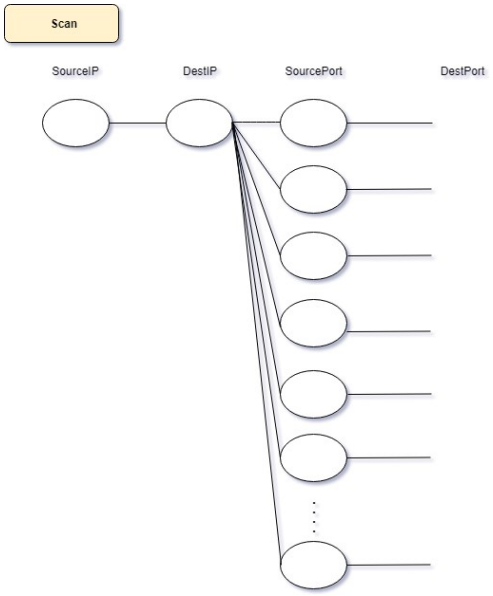

Fig. 23. Scan Pattern

A flow is characterized as P2P according to the port history if all the distinct ports connecting to an (IP, port) pair are used by common services, i.e., ports lower than 1024, and if the pair appears in at least ten flows (Figure 24).

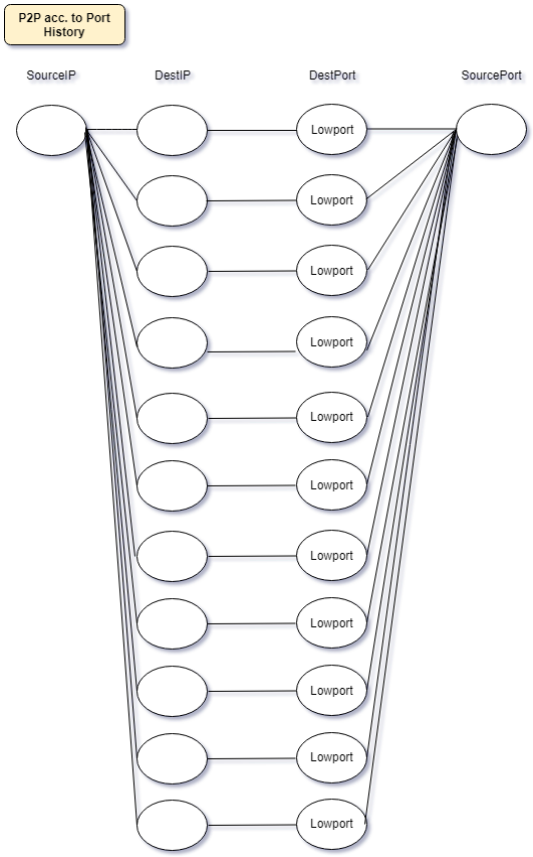

Fig. 24. Port History Pattern

3.9 SNA and Visualization

The approaches used in the previous sections lead to the assignment on flow level of specific services after their identification by using the known characteristics of the network such as port number, transfer size or protocol. This analysis facilitates an accurate finding of the common services, anomalies and P2P traffic and represents a good basis for identifying further patterns within the network by means of SNA. Following sections contain the description of the SNA methods applied on both data sets and how these methods characterize the network traffic.

3.9.1 Introduction to SNA

According to [25, p.278], the basis of SNA is illustrating the social network as a sociogram consisting of nodes and relations. In a social network, an actor can be for instance a person, an organization, a blog or a host and is represented as a node. A relation represents a linkage or an edge between the actors and can have different connotations: acquaintance, friendship, exchange, data transfer, etc. Relations are symmetrical in case of both data sets (each node sends and receives to/from another node). The focus is set not on the individuals and their attributes but on their structure and the relationship between them. The benefit of such a representation is that it allows the analysis of social processes as a result of relations between social entities. SNA uses two kinds of tools to represent information about patterns of ties among these actors: graphs and matrices. Representing data as a matrix is a pre-step for data manipulation and utilization of further methods. The main data frame created in the previous steps is the fundamental matrix from which several other data frames serving the purpose of graph visualization and analysis are built. According to [13, p.167], SNA addresses one important feature of social structure: the provenience and dispersion of power, where power is seen not as a node attribute, but as a result of the relations between nodes.

A network graph visualization can suggest some of the most important characteristics of the whole network structure, for instance, if all nodes are connected, if there are many or few ties among the actors, if any subgroups or local clusters exist or if some actors have many or a few ties when compared with other actors. According to [12, p.6], the analysis of Internet traces facilitates the creation of different network graphs that rely upon the manner in which the data is aggregated. The underlying graphs of the two data sets are undirected, since each node within a pair of nodes sends and receives to and from each other.

This thesis aims to build a bridge between social science and network flow analysis by finding the network behavioral similarities between them and interpreting the concepts used in social analysis by means of network traffic. For instance, terms as “friendship” or “political adherence” from the social field are translated into concepts as “communicates to” or “is a subnet of” of the network traffic. Popularity, for instance, refers to the number of connections an IP has with other IPs.

Out of the software that facilitate such social network specific analysis, R was chosen, since it offers a large range of packages and methods, which are devoted to the features of network analysis: standard manipulation, characterization of network data, network modelling and visualization [30, p.8]. Another reason for choosing R is that a network manager/operator can have a single self-configurable script, which can be used to perform at once the whole analysis of the network data: from data processing and pre-processing to the identification of common services, anomalies and P2P traffic and statistical SNA.

The result of applying social network methods on the network data is the emergence of several new dimensions (columns) within the main data frame, which contain the outcomes of this analysis. These results are correlated and compared to each other and with the initial findings in order to find additional communication patterns within the network except those derived from the usage of standard ports and/or protocols. In other words, the SNA methods are used to see if it possible to profile a network only by studying its dynamics.

3.9.2 Popularity and Network Behavior

The pre-filtering step of traffic analysis contains the analysis of the popularity of a host, which is defined as its interactions with other hosts [31, p.234]. It can be differentiated between the following interaction types: one-to-one (download), one-to-several (browsing) and one-to-many (scans) [32, p.321]. Since the aim is to map the results on flow level into the main data frame, six new columns are added during this process. Three out of these columns describe the popularity measures described above for the source IP of the flow and the other three define the popularity of the destination IP of the flow. IP addresses connected to only one IP address are profiled as "one-to-one:download" into the main data frame. The threshold used for the identification "one-to-several" and "one-to-many" interactions is defined as the total duration of the flows (in seconds) divided by 3. The value 3 is chosen as a threshold since it is very probable that if the traffic amount assigned to an IP address, which connects to other IP addresses, exceeds a third of the total duration of the flows in seconds, then this particular traffic can be labeled as scan. Thus, if an IP address sends at least one packet to more than two other IP addresses in the trace and to less IP addresses than the threshold value, then this IP address is labeled as "one-to-several:browsing" into the main data frame. If the number of IP addresses connected to one IP address exceeds the threshold value, then this IP address is labeled as "one-to-many:netscan" into the main data frame.

The number of ports divided by the number of peers define the network behavior in terms of attacks (port scans) or normal behavior. Usually, servers reply on a single port in case of standard protocols and clients use a distinct random port for connecting to a server. Many connections initiated may indicate attacks or port scans. Thus, a similar methodology with the one used for profiling the popularity of a host is applied, by labeling as attacks all flows for which the number of ports divided by the number of peers exceeds the total duration of the flows divided by 3. On the contrary, if this value is below the threshold, then the flow is labeled as normal behavior.

3.9.3 Properties of the Network and its Actors

Customarily, graphs are mathematical structures containing a group of vertices (nodes, actors) and a set of edges (links, connections) and can be labeled accordingly as $G = (V, E)$. The edges E are unordered pairs (u, v) of separate vertices $u, v \in V$

[30, p.14]. A graph-based framework is embraced for plotting the relational data in network analysis. For that reason, the R package "igraph" is used to create the network graph and to extract and outline its characteristics.

The network is "decorated" by labeling the edges and the vertices with its attributes, i.e., values from the main data-frame associated with the corresponding graph. The vertex attributes are both discrete (for instance the protocol, port, IP address) and continuous (transfer size, betweenness centrality, closeness, etc.). The vertex attributes can also mean assigning colors to the vertices in course of the analysis, so that they can be differentiated from other vertices.

So that all dimensions of the network traffic can be analyzed, a vertex can represent different elements of the network:

- IP address as a vertex: for measuring the way in which hosts communicate with each other.
- Ports as a vertex: for measuring the way in which the endpoints of communication are linked with each other.
- (IP, port) pair as vertex: for measuring the way types of services assigned to a host communicate with each other.
- (IP, port) and IP as vertex: for measuring the way types of services assigned to a host communicate with all the other hosts.
- (IP, port) and port as vertex: for measuring the way types of services assigned to a host communicate with the endpoints of communication.

The classification outlined above results in 5 dimensions of the network, where the following methods are applied on each of the corresponding graphs: neighbors, egocentric networks, degree, betweenness centrality, closeness centrality, graph components (clusters), communities detection by means of a greedy algorithm[23] and functional classes. The results of these methods are analyzed in parallel, mapped into the main data frame on flow level and correlated between each other with the intent of finding similarities between them and the communication patterns outlined in Section 3.8.

Since in R not only the network graphs are encoded in data frames but also the attributes, the vertices and the edges are represented in a separate data frame. The first column of the vertex data frame includes the vertex names and all the other columns contain the values of the vertex attributes. The edge list that defines the graph is contained in the first two columns of the edge data frame, while all the following columns consist of the edge attributes [30, p.16].

Each vertex can be defined by a variable that is based on its relational features, such as the numbers of ties to other vertices**.** Adjacency is the elementary notion of the graph connectivity, which indicates how many vertices need to be removed so that

[23] https://en.wikipedia.org/wiki/Greedy_algorithm

one actor becomes unreachable from another actor. Two nodes are adjacent or neighbors if they are connected by an edge, while two edges are adjacent if a common vertex links them. The local structure of a given vertex can be highlighted in form of an egocentric network visualization, which is a sample of a network's local area [14, p.9] and shows the vertex (ego), its immediate neighbors, and all edges among them. The neighborhoods are the number of distinct vertices directly connected with the center. According to [14, p.9], the egocentric network highlights the opportunities and restraints of the ego, as a result of its embedment within the network. If the vertices connected to the ego are not directly connected to one another, then the ego has the role of a broker [13, p.135], since it is the center of a "star" network, being placed on a path between the other vertices.

The number of edges incident on a vertex are referred as the degree of a vertex, which is an index of connectivity in the network [30, p.22]. In a social context, the degree values can signify for instance the number of friendship ties among a class and reveals the actor's prominence in the network [29, p.59]. There is a high probability that communication occurs and that the vulnerability of the connection decreases if there are many flows between a pair of vertices. Therefore, actors who have more ties (higher degrees) and egos who bridge gaps between the vertices may be advantaged. Both neighborhood and degree can give insight into the popularity extent and power [14, p.146] of a particular vertex and the degree histogram can reveal important information, by highlighting the degree distribution within the network.

Betweenness centrality is a measure that summarizes the extent to which a vertex is situated between other pair of vertices and implies that "importance" depends on the node position with regard to the paths inside the network [30, p.48]. Betweenness centrality can reveal which node plays a key role in facilitating the direct flow of information between itself and other nodes within the network. In contrast with betweenness centrality, closeness centrality defines how close a node is to the other nodes and is defined by [43] as the inverse of the total distance from one particular node to all the other nodes.

The network cohesion is characterized by the coherence magnitude of the vertex subsets and is defined by measures such as graph components and communities. The differences in the manner that actors are enclosed in the structure of groups outline the behavior that these actors are inclined to practice and how they see their "society" [13, p.171]. The number and size of groups including their underlying connection patterns within the network can be used to conclude the restraints of the actors and the development of the graph itself [13, p.171]. A connected component is a "maximally connected subgraph" [30, p.57] and can provide insight into how structured is the graph. Large components indicate that the graph is rather dense, where density can give insight into the development of the speed at which information propagates among the nodes. [33, p.267]

The graph is partitioned into communities based on a hierarchical clustering method used in data analysis. This approach uses a greedy approach for examining the space of all available partitions, by consecutively changing subsequent candidate

partitions either by merging or by splitting them. At each step, the current candidate partition is changed in a manner that minimizes a stated cost measure. A well-known cost measure is modularity, which is defined by [34, p.7] as:

$$Q = \sum_i \left(e_{ij} - a_i^2\right) = \mathrm{Tr}\, e - \|e^2\|,$$

where e_{ij} is the chunk of all edges that connect vertices in community i to vertices in community j, a_i is either the row or the column sums $\sum_j e_{ij}$, $\mathrm{Tr}\, e$ is the trace of the symmetric matrix with the elements e_{ij} that defines the fraction of edges linking vertices within the same community. The quantity Q is the difference between the group of edges connecting vertices alike and the predicted value of the same quantity Q yielding the same community partitions and random links between the vertices [34, p.7]. Large values of modularity signify that the given candidate partition has an uncommon group structure and can eventually occur under randomizing the allocation of vertices. Since the optimization of modularity depends over searching all possible network partitions and thus it is expensive in large-sized networks, a fast and greedy approach for optimization is used, as proposed by [35]. This ideas can be implemented using the function fastgreedy.community of the R library "igraph" and the outcome is an item of the class "communities", which can be further used for other functions. The result of this agglomerative hierarchical clustering algorithm can be represented in form of a tree, called a dendrogram, which facilitates the identification of approximate equivalence classes [13, p.213]. Since it is commonly expected that the existence of cohesive fractions of vertices assumes also some similarity in particular relevant vertex attributes, each attribute is validated against its community (functional classes). Graph partitioning can be seen as a tool for identifying subsets of vertices without knowing a priori the vertex attributes.

3.9.4 Similarity Analysis

Analyzing data in terms of equivalence classes enables the generalizations about social structure and behavior. In order to do that, actors should be regarded not as unique individuals, but as categories, which emerge by grouping the actors that are most similar [13, p.196]. Similarity refers to the patterns of relations between actors and not to their underlying attributes, which emphasizes the concept of "social role" or "social position" [13, p.196]. This thesis addresses the idea of structural equivalence, which assumes that two actors are structurally equivalent if they have the same relationships to all other actors [13, p.199]. The notion of relationships or link of a host is understood in terms of the host's total neighbors, since it is appropriate to analyze the distinct connections of a host rather than measuring all of the connections, which is defined by degree – see Section 3.9.3. This phase of the analysis contains two steps:

– Identify hosts that are accurately structural equivalent to each other, thus have the same neighbors.
– Identify pair of hosts that are approximately comparable to one another, by using the convergence of iterated correlations and grouping them within equivalence blocks.

The results of this analysis show which host can be substituted by another host that has the same links to other hosts, respectively which hosts have the same positions within the network with respect to all other hosts. The hosts that are either exactly structurally equivalent or roughly structurally equivalent deal with the same constraints and opportunities [13, p.202].

3.9.5 Temporal Analysis

The analysis of the network behavior over time gives insight into the most "busy" or critical time interval. For this purpose, the metrics of betweenness, degree, neighbors and bytes transfer are visualized over the whole time span of the data log. This approach facilitates a further investigation of the assets that are responsible for an unusual traffic activity, by considering the peaks (local maxima) of the temporal plots.

4 Explorative Analysis of Data Set I

The methods and algorithms described in Section 3 are applied on both data sets, by means of the R script. The traces of the first data set were collected from the perimeter of an enterprise network. The raw data file contains 13263 anonymized flows and captures the traffic occurring in a period of approximately 5 minutes. The file is converted to a data frame within the R environment, for the intent of a further analysis. Some of the attributes of the data frame are already given (the initial raw format) and others arise from the locations of the actors in the network and from analysing the underlying ports, bytes and protocols. The column headers are renamed from the conventional NetFlow labels to short names for improving the data understanding. As a result of the pre-processing and processing step, the data frame is compressed to 1886 rows, where each row represents a bidirectional flow. The result of identifying anomalous traffic consists in some flows containing an invalid type of service flag and three flows flagged in the "Worms_Trojans" column as "/Worm.Opasoft, W32.Opaserv.Worm". This profiling is summarily represented in a data frame, which contains the suspicious 5-tuple flows (Figure 25).

SourceIP	DestIP	SourcePort	DestPort	Protocol	SuspectFlow	DoS	Worms_Trojans
146.17.26.0	146.16.239.120	57064	81	TCP	invalid_Tos		
146.17.26.0	146.16.239.120	57086	81	TCP	invalid_Tos		
146.17.26.0	146.16.239.120	57097	81	TCP	invalid_Tos		
146.17.26.0	146.16.239.120	57105	81	TCP	invalid_Tos		
146.17.26.0	146.16.239.120	57107	81	TCP	invalid_Tos		
146.17.26.0	146.16.239.120	57117	81	TCP	invalid_Tos		
146.17.26.0	146.16.239.120	57119	81	TCP	invalid_Tos		
146.17.26.106	146.16.239.121	58820	81	TCP	invalid_Tos		
146.17.26.18	146.16.251.34	137	137	UDP			/Worm.Opasoft, W32.Opaserv.Worm
146.17.26.6	146.16.239.120	56230	81	TCP	invalid_Tos		
146.17.26.6	146.16.239.121	56303	81	TCP	invalid_Tos		
146.17.26.80	146.16.239.121	62576	81	TCP	invalid_Tos		
146.31.247.70	146.17.26.18	137	137	UDP			/Worm.Opasoft, W32.Opaserv.Worm
146.31.247.70	146.17.26.92	137	137	UDP			/Worm.Opasoft, W32.Opaserv.Worm

Fig. 25. Anomalous Flows in Network I

The tuples (146.17.26.18, 146.16.251.34, 137, 137, UDP), (146.31.247.70, 146.16.26.18, 137, 137, UDP), (146.31.247.70, 146.16.26.92, 137, 137, UDP) contain the patterns of "Worm.Opasoft, W32.Opaserv.Worm": port 137, protocol UDP and 78 transfer bytes. Although it is recommended to monitor these hosts more carefully since they are identified as generating malicious activity, also a false interpretation as worm activity is possible. It is probable that this trace is generated by a failure of the host's DNS queries due to a configuration error, which further produces a flood of name resolution attempts by NetBIOS over the IP protocol, as outlined in [36]. A defense recommendation is the blocking of ports 135-139 over TCP and UDP and of port 445 at the firewall. If access to the servers over WAN is required, then the connection should be limited to trusted hosts, which should be further monitored [36].

The traffic volume of the top protocols within the network can be visualized in Figure 26. The chart indicates that almost all the traffic happens over TCP/UDP and ESP and ICMP cover only 3% of the whole traffic.

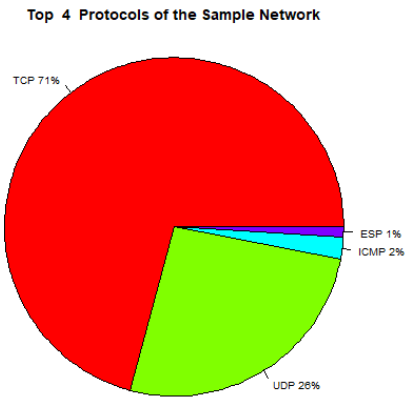

Fig. 26. Top Protocols by Traffic Volume of Network I

In order to profile the most requested services by the client within the sample network, the outgoing traffic is sorted by the occurrence of the destination ports. As expected, the most requested service is web (81, 443), followed by DNS (53) and LDAP (389) (Figure 27). The servers are expected to manage the same type of services.

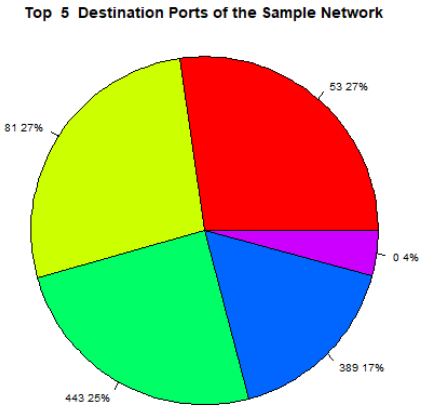

Fig. 27. Top Services of the Network I

For profiling the clients within the network (clients are defined as the hosts which initiate the communication), the outgoing traffic is sorted by the occurrence of the source ports. The result of this profiling is visible in Figure 28. The most used client service is LDAP[24] (port 3268), which is by definition insecure.

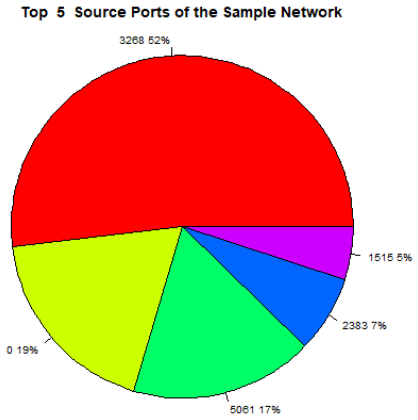

Fig. 28. Top Clients of the Network I

Following tables (Figure 29, Figure 30, Figure 31 and Figure 32) show the outcome of the profiling of the common services within the network, as illustrated in Section 3.6. These tables can be found in the HTML report, as well.

The web servers responsible for more than 1% of the transfer bytes are almost all within the same prefix (146.16.x.x), meaning that they all belong to the same subnetwork (Figure 29).

DestIP	DestPort
146.31.171.70	443
146.16.253.18	443
146.16.239.124	80
146.16.239.121	81
146.16.239.120	81
146.16.224.49	80
146.16.224.100	443
146.16.224.100	80

Fig. 29. Top Web Servers of the Network I

[24] https://en.wikipedia.org/wiki/Lightweight_Directory_Access_Protocol

All the web clients sending more than 1% of the transfer bytes use more than one port assigned to the web traffic and are all within the same prefix (146.17.26.x), meaning that they are all part of the same subnetwork (Figure 30).

SourceIP	DestPort
146.17.26.92	443
146.17.26.92	80
146.17.26.92	81
146.17.26.89	443
146.17.26.89	80
146.17.26.89	81
146.17.26.80	443
146.17.26.80	80
146.17.26.80	81
146.17.26.6	443
146.17.26.6	81
146.17.26.6	80
146.17.26.28	443
146.17.26.28	81
146.17.26.28	80
146.17.26.23	443
146.17.26.23	80
146.17.26.118	443
146.17.26.118	80
146.17.26.118	81

Fig. 30. Top Web Clients of the Network I

Figure 31 illustrates the only DNS server within the network, which resides on the same subnetwork assigned to the majority of the web servers (146.16.x.x).

DestIP	DestPort
146.16.240.194	53

Fig. 31. Top DNS Servers of the Network I

Most of the DNS clients in Figure 32 are also web clients (Figure 30) and they belong to the same subnetwork assigned to the web clients (146.17.26.x).

SourceIP	DestPort
146.17.26.92	53
146.17.26.83	53
146.17.26.80	53
146.17.26.6	53
146.17.26.28	53
146.17.26.25	53
146.17.26.23	53
146.17.26.22	53
146.17.26.18	53
146.17.26.13	53
146.17.26.125	53
146.17.26.118	53
146.17.26.109	53
146.17.26.106	53
146.17.26.0	53

Fig. 32. Top DNS Clients of the Network I

There are no mail and remote services running within the network, which may indicate that the traffic was captured outside working hours.

The P2P algorithm detected 285 P2P flows out of 1886 total flows, which means that roughly 15% of the total traffic is P2P traffic. All the flows are flagged as having a normal behavior according to the procedure outlined in the Subsection 3.9.2 and 96,67% of the flows are flagged as “one-to-several:browsing”, while the rest is flagged as “one-to-one:download”. No flow is marked as “one-to-many:netscan” or as “attack”. This indicates that neither download, nor scanning is prevalent within the network.

Recall that the meaning of a vertex is different in particular contexts of research and this leads to a parallel analysis of five types of network dimensions. Firstly, the network is built around the IP addresses that are represented as a vertex (referred further as IP Network). Secondly, the dynamics between the ports as a vertex is analyzed (referred further as Port Network). Thirdly, the (IP, port) pairs are compressed to a vertex, with the intent of measuring how the types of services assigned to a host communicate with each other (referred further as Pair Network). Fourthly, the (IP, port) pairs and the IP addresses are labeled as a vertex (referred further as PairToIP Network). Fifthly the (IP, port) pairs and the port form a vertex (referred further as PairToPort Network).

The network is analyzed in parallel along the five heuristics above with the intent of finding similarities between the outcomes of the measures applied.

The attributes corresponding to a given vertex are mapped in a data frame, which is used further to build the network graph. Depending on the way a vertex is constructed, also the attributes may be different. For instance, when mapping the ports to a vertex, there is only the host type (in terms of P2P or not P2P) that can be determined as an attribute of the vertex, since the other attribute types cannot be used as unique identifiers. Figures 33, 34 and 35 illustrate fractions of the three data frames –

IP as a vertex, IP and port (pair) as a vertex, port as a vertex - populated with the vertex attributes.

ID	Label	ClientServer	RemoteClientServer	Mail	DNS	TotalTransfer	SuspectFlow	DoS	Worms_Trojans	possibleScan	VPN	HostType
87	146.16.238.116	host	host	host	host	20192.6638	NA	NA	NA	NA	VPN	NONP2P
88	146.16.238.96	host	host	host	host	1070.0000	NA	NA	NA	NA	NA	NONP2P
89	146.16.224.100	webServer	host	host	host	122372.2627	NA	NA	NA	NA	VPN	NONP2P
90	146.16.224.23	host	host	host	host	5109.8256	NA	NA	NA	NA	VPN	NONP2P
91	146.16.224.49	webServer	host	host	host	16900.3790	NA	NA	NA	NA	VPN	NONP2P
92	146.16.239.120	webServer	host	host	host	91044.1502	invalid_Tos	NA	NA	NA	VPN	NONP2P
93	146.16.239.121	webServer	host	host	host	145948.1282	invalid_Tos	NA	NA	NA	VPN	NONP2P
94	146.16.239.123	host	host	host	host	7956.0579	NA	NA	NA	NA	VPN	NONP2P
95	146.16.239.124	webServer	host	host	host	9604.7838	NA	NA	NA	NA	VPN	NONP2P
96	146.16.242.12	host	host	host	host	9586.6284	NA	NA	NA	NA	VPN	NONP2P
97	146.16.249.140	host	host	host	host	1212.6063	NA	NA	NA	NA	NA	NONP2P
98	146.17.26.105	host	host	host	host	1396.9776	NA	NA	NA	NA	NA	P2P
99	146.16.230.97	host	host	host	host	2824.8486	NA	NA	NA	NA	VPN	NONP2P
100	146.16.230.98	host	host	host	host	4526.5293	NA	NA	NA	NA	VPN	NONP2P
101	146.17.26.107	host	host	host	host	2347.8743	NA	NA	NA	NA	NA	P2P
102	146.16.225.94	host	host	host	host	2878.0510	NA	NA	NA	NA	VPN	NONP2P
103	146.16.251.34	host	host	host	host	350.0000	NA	NA	/Worm.Opasoft, W32.Opaserv.Worm	NA	NA	NONP2P
104	146.16.227.4	host	host	host	host	2413.0545	NA	NA	NA	NA	VPN	NONP2P
105	146.16.226.213	host	host	host	host	875.2222	NA	NA	NA	NA	NA	NONP2P
106	146.16.238.115	host	host	host	host	2765.1190	NA	NA	NA	NA	NA	NONP2P
107	146.16.249.157	host	host	host	host	7241.7641	NA	NA	NA	NA	VPN	NONP2P

Fig. 33. Vertex Attributes of the IP Network I

ID	Label	ClientServer	RemoteClientServer	Mail	DNS	TotalTransfer	SuspectFlow	DoS	Worms_Trojans	possibleScan	VPN	HostType
53	146.17.26.88 0	webClient	remoteHost	mailHost	DNSHost	164.0000	NA	NA	NA	NA	VPN	NONP2P
54	146.17.26.89 0	webClient	remoteHost	mailHost	DNSHost	300.9252	NA	NA	NA	NA	VPN	NONP2P
55	146.17.26.92 0	webClient	remoteHost	mailHost	DNSClient	2481.6350	NA	NA	NA	NA	VPN	NONP2P
56	146.16.251.99 5061	webHost	remoteHost	mailHost	DNSHost	7138.3323	NA	NA	NA	NA	NA	P2P
57	146.16.250.103 23018	webHost	remoteHost	mailHost	DNSHost	408.0000	NA	NA	NA	NA	NA	P2P
58	146.16.250.103 23019	webHost	remoteHost	mailHost	DNSHost	180.0000	NA	NA	NA	NA	NA	P2P
59	146.17.25.33 43397	webHost	remoteHost	mailHost	DNSHost	283.0000	NA	NA	NA	NA	NA	NONP2P
60	146.16.250.103 31354	webHost	remoteHost	mailHost	DNSHost	408.0000	NA	NA	NA	NA	NA	P2P
61	146.16.250.103 31355	webHost	remoteHost	mailHost	DNSHost	180.0000	NA	NA	NA	NA	NA	P2P
62	146.17.25.36 49760	webHost	remoteHost	mailHost	DNSHost	283.0000	NA	NA	NA	NA	NA	NONP2P
63	146.16.120.111 5012	webHost	remoteHost	mailHost	DNSHost	428.0000	NA	NA	NA	NA	NA	P2P
64	146.16.120.111 5013	webHost	remoteHost	mailHost	DNSHost	196.0000	NA	NA	NA	NA	NA	P2P
65	146.17.25.43 47388	webHost	remoteHost	mailHost	DNSHost	283.0000	NA	NA	NA	NA	NA	NONP2P
66	146.17.25.48 32948	webHost	remoteHost	mailHost	DNSHost	283.0000	NA	NA	NA	NA	NA	NONP2P
67	146.17.223.220 5014	webHost	remoteHost	mailHost	DNSHost	428.0000	NA	NA	NA	NA	NA	P2P
68	146.17.223.220 5015	webHost	remoteHost	mailHost	DNSHost	196.0000	NA	NA	NA	NA	NA	P2P
69	146.17.25.62 60409	webHost	remoteHost	mailHost	DNSHost	283.0000	NA	NA	NA	NA	NA	NONP2P
70	146.17.223.154 5010	webHost	remoteHost	mailHost	DNSHost	428.0000	NA	NA	NA	NA	NA	P2P
71	146.17.223.154 5011	webHost	remoteHost	mailHost	DNSHost	196.0000	NA	NA	NA	NA	NA	P2P
72	146.16.250.119 23454	webHost	remoteHost	mailHost	DNSHost	408.0000	NA	NA	NA	NA	NA	P2P
73	146.16.250.119 23455	webHost	remoteHost	mailHost	DNSHost	180.0000	NA	NA	NA	NA	NA	P2P
74	146.17.25.82 45529	webHost	remoteHost	mailHost	DNSHost	283.0000	NA	NA	NA	NA	NA	NONP2P

Fig. 34. Vertex Attributes of the Pair Network I

ID	Label	TotalTransfer	HostType
9	5012	2120.0000	P2P
10	5013	964.0000	P2P
11	47388	283.0000	NONP2P
12	32948	283.0000	NONP2P
13	5014	856.0000	P2P
14	5015	392.0000	P2P
15	60409	283.0000	NONP2P
16	5010	1264.0000	P2P
17	5011	572.0000	P2P
18	23454	408.0000	P2P
19	23455	180.0000	P2P
20	45529	283.0000	NONP2P
21	68	4088.6667	NONP2P
22	60568	185.8182	NONP2P
23	60569	152.6000	NONP2P

Fig. 35. Vertex Attributes of the Port Network I

To provide a network overview regarding its structure and the services used, the network can be visualized from the perspective of different attributes labeled with different colors within the network graph. For this purpose, the IP Network is used as a basis for the visualizations, since it is the only one among all five network types that emphasizes the connections among hosts in form of edges. The network visualization yields 2 clusters, one containing the majority of the nodes and the other one including a few vertices centered around one vertex in form of a star. The larger cluster contains one graph component on the left side of the cluster, which is delimited from the right side of the cluster by a sort of bridge.

Figure 36 provides a network visualization from the point of view of client/server interaction, where the clients have the color blue, the servers are green and the rest of the hosts are red. As noticeable in the Figure 29 and in Figure 30, the web servers and the web clients are mainly included in a single subnetwork, which is also visible in the component forming the left side of the graph in Figure 36, which contains all the web servers and all the web clients.

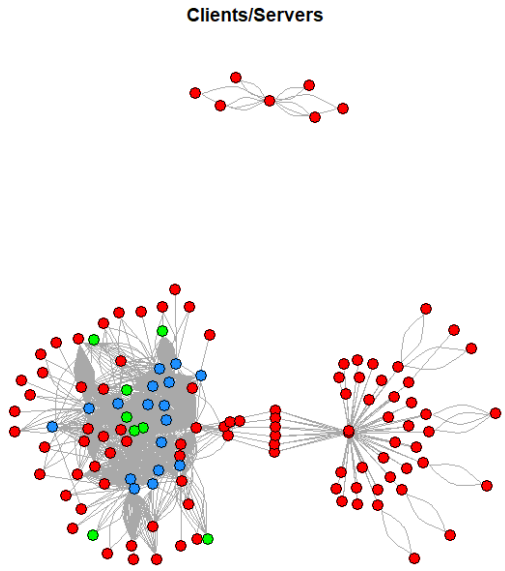

Fig. 36. Network I Visualization Clients/Servers

Figure 37 provides a network visualization from the perspective of DNS interactions, where the DNS clients have the color blue, the DNS servers are green and the rest of the hosts are red. Comparing this visualization with the one in Figure 36, it is visible that the web clients overlap with the DNS clients. As expected, the DNS server is situated at the “entrance” of the graph component on the left side of the network, which is delimited from the right side of the network by a “bridge” being assigned to none of the common services. As noticeable in the Figure 31 and in Figure 32, the DNS servers and the DNS clients are mainly included in a single subnetwork, which is also visible in Figure 37, which contains all the DNS servers and all the DNS clients in the network component located on the left side of the graph.

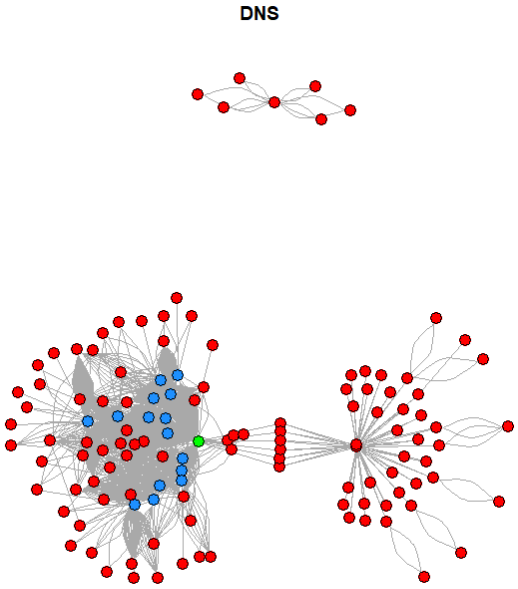

Fig. 37. Network I Visualization DNS

Since there are no mail services within the network, the vertices have only one color (Figure 38).

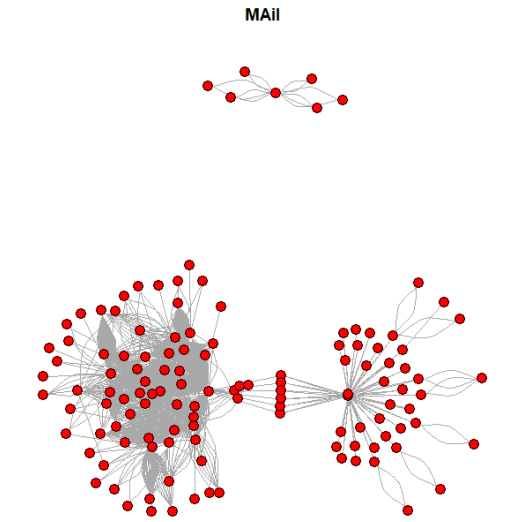

Fig. 38. Network I Visualization Mail

Figure 39 provides a visualization of the network from the point of view of P2P interaction; the P2P hosts have the color red, the non-P2P hosts are blue and the hosts that are both P2P and non-P2P are green. It is visible that the cluster in the upper part of Figure 39 contains exclusively P2P traffic, while the cluster below has P2P nodes split mostly on the edge of the graph. The hosts that are both P2P and not P2P extend across both components of the big cluster and are spread in form of a star in the component on the right side of the cluster.

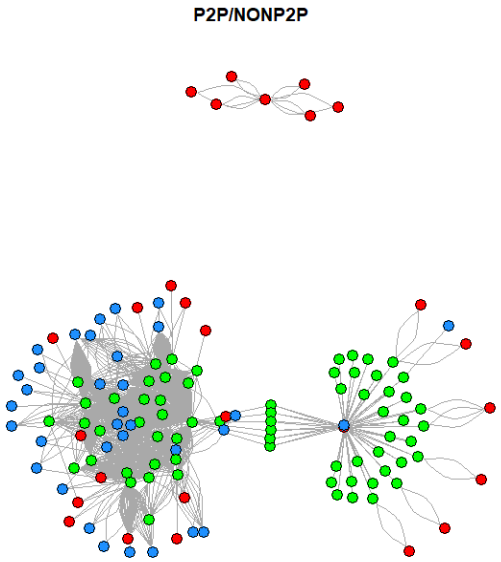

Fig. 39. Network I Visualization P2P Traffic

Particular information that is desiderated to be communicated by a network visualization might advise that only an appropriate subgraph is shown. For instance, it is practical to focus on the structure restricted to the immediate area of a given vertex by means of egocentric network visualizations. The largest egocentric subgraphs containing the first-order neighborhoods of the IP network can be visualized in the Figures 40, 41, 42, 43 and 44. The nodes are labeled with the vertex ID and the color encoding is as follows: DNS client – blue, DNS server – green, mail Client – violet, mail Server – violet, client web – yellow, web server – orange, host – red. The mapping of the vertex ID to the corresponding IP address can be found in the data frame containing the attributes of the IP addresses, whose header is outlined in Figure 33.

The biggest egocentric subgraph (Figure 40) is the one surrounding the vertex having the ID 56, which together with the vertices connected to it belong to none of the common services identified and marked as non-P2P in the main data frame.

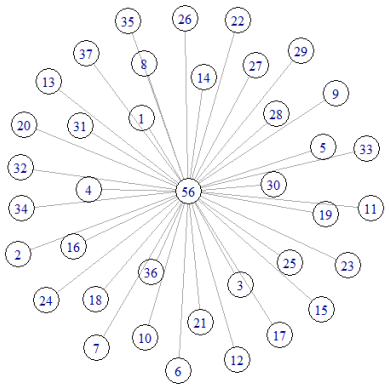

Fig. 40. First Order Neighborhood Network I - Rank 1

The next largest egocentric subgraph (Figure 41) encircles again a host with the same connection patterns as the node 56: all hosts are assigned to no common services.

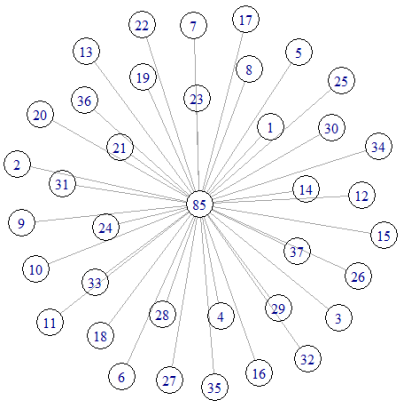

Fig. 41. First Order Neighborhood Network I - Rank 2

The center of the next egocentric subgraph (Figure 42) contains a host acting as both web client and DNS client. The neighbors of this vertex are a DNS server, four web servers and hosts being assigned to no particular service.

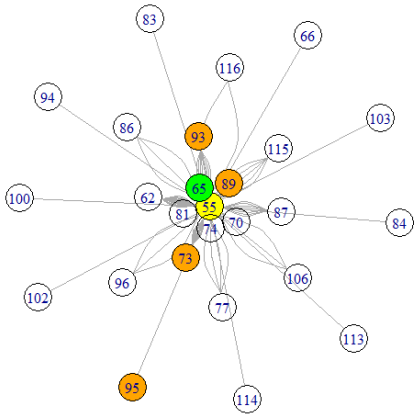

Fig. 42. First Order Neighborhood Network I - Rank 3

The following egocentric subgraph (Figure 43) has its focal point a DNS server, surrounded by DNS clients, web clients and hosts using no common services.

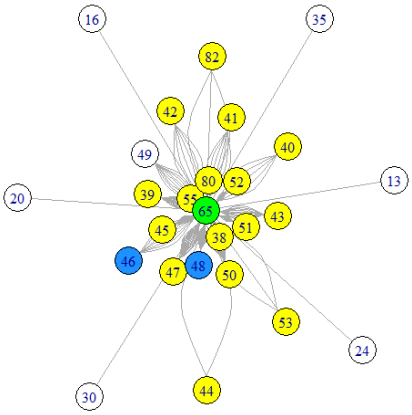

Fig. 43. First Order Neighborhood Network I - Rank 4

The fifth largest neighborhood belongs to a web server that is surrounded by web clients, DNS clients and one node belonging to no service (Figure 44).

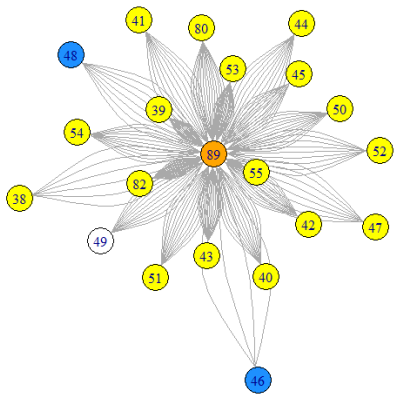

Fig. 44. First Order Neighborhood Network I - Rank 5

Comparing the plots in the Figures 40, 41, 42, 43 and 44 to the visualization of the full network illustrated in Figures 36, 37, 38 and 39, it is clear that the subnetworks in Figure 40 and Figure 41 capture the majority of the component on the right side of the larger cluster (red vertices which stand for no common service, respectively possible P2P traffic), while the subnetworks in Figures 42, 43 and 44 capture the majority of the component on the left side of the big cluster (vertices having different colors which stand for different services). As expected, the DNS servers and the web servers belong to the top 5 biggest egocentric subnetworks, which validates their character of transmitting data or being accessed from many web hosts.

The top biggest egocentric subgraphs of the Port Network, Pair Network, PairToIP Network and PairToPort Network are summarized in form of a table, since the high amount of neighbors makes the corresponding graph visualization unclear.

The largest egocentric subgraph of the Port Network contains the port 81 (ID 1.479) and its 382 direct neighbors (Figure 45). Port 81 is HTTP, which explains its high amount of neighbors and validates the findings from the analysis of the top ego-centric subgraphs of the IP Network, where a host labeled as web server is in top 5 biggest first-order neighborhoods.

NeighborsCount	ID
382	1479
378	1472
346	1476
235	1481
166	181

Fig. 45. First Order Neighborhood Port Network I

The largest egocentric subgraph of the Pair Network contains the pair (146.16.240.194, 53) (ID 1.626) and its 384 direct neighbors (Figure 46). This pair defines the only DNS server of the network, profiled in the Section 3.6.4 and identified in the top five largest first-order neighborhoods of the IP Network.

NeighborsCount	ID
384	1626
276	1663
230	1668
153	1667
84	1683

Fig. 46. First Order Neighborhood Pair Network I

The largest egocentric subgraph of the PairToIP Network contains the IP address 146.16.240.194 (ID 2.047) and its 412 direct neighbors (Figure 47). This IP address is the DNS server, which is identified also by the heuristics before.

NeighborsCount	ID
412	2047
277	2071
230	2075
153	2074
148	2063

Fig. 47. First Order Neighborhood PairToIP Network I

The largest egocentric subgraph of the PairToPort Network contains the port 53 (ID 3.454) and its 384 direct neighbors (Figure 48), which is assigned to the DNS service.

NeighborsCount	ID
384	3454
382	3461
378	1626
349	3458
275	1663

Fig. 48. First Order Neighborhood PairToPort Network I

The degree distribution of a network is a rescaling of the degree frequency and is described as the collection $\{f_d\}_{d\geq 0}$, where f_d is the vertex v set having the degree $d_v = d$ [30, p.44]. The strength distribution is a rescaling of the strength frequency and is calculated by summing the edge weights incident to a particular vertex [30, p.44]. Both measures can be represented in form of a histogram and they can provide valuable information about the network, since the preponderance of high orders of magnitude may indicate an attack or a scan.

The histograms of all five network types look alike (Figures 49, 50, 51, 52, 53), both regarding the degree and the strength distribution. The majority of the vertices have a quite low degree (between 0 and 50), but there is also a small fraction of vertices having continuously higher orders of magnitude (between 50 and 200).

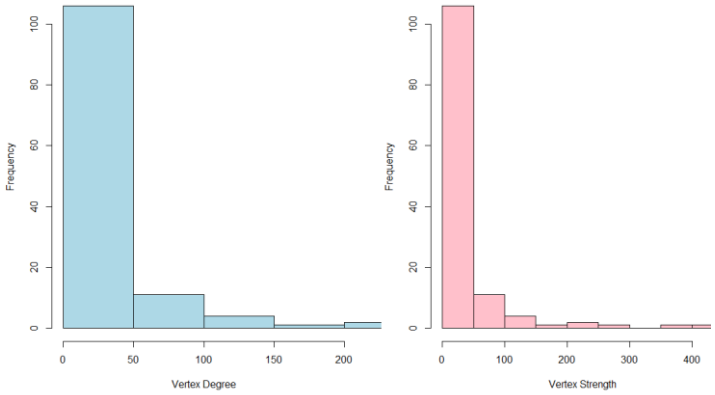

Fig. 49. Degree Distribution (left) and Strength (right) of IP Network I

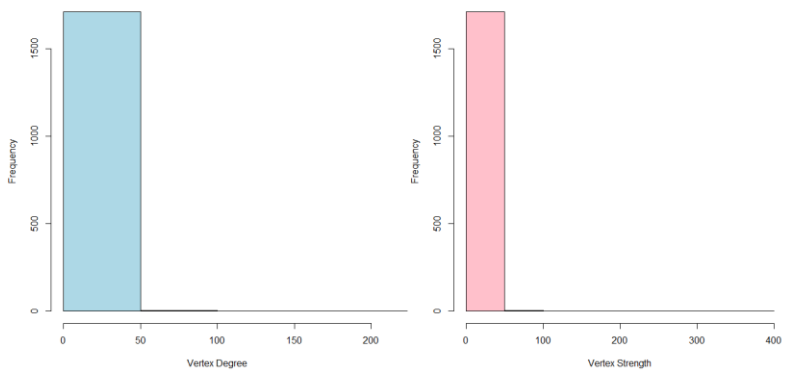

Fig. 50. Degree Distribution (left) and Strength (right) of Port Network I

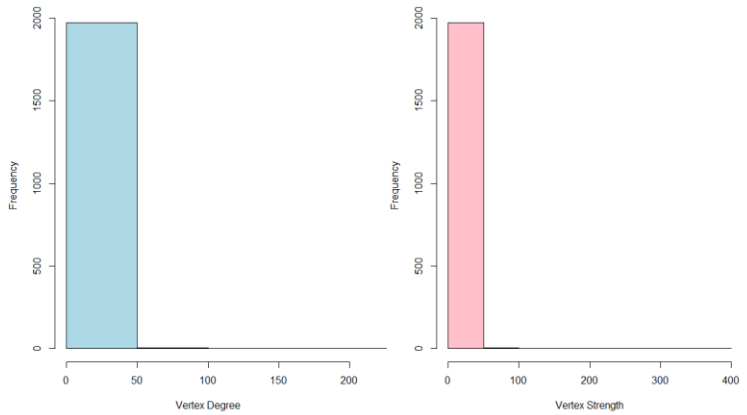

Fig. 51. Degree Distribution (left) and Strength (right) of Pair Network I

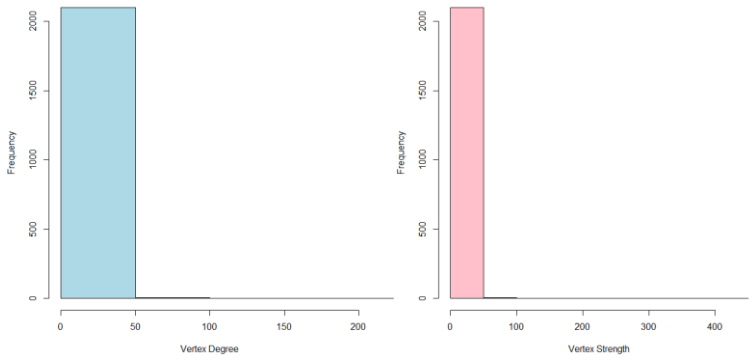

Fig. 52. Degree Distribution (left) and Strength (right) of PairToIP Network I

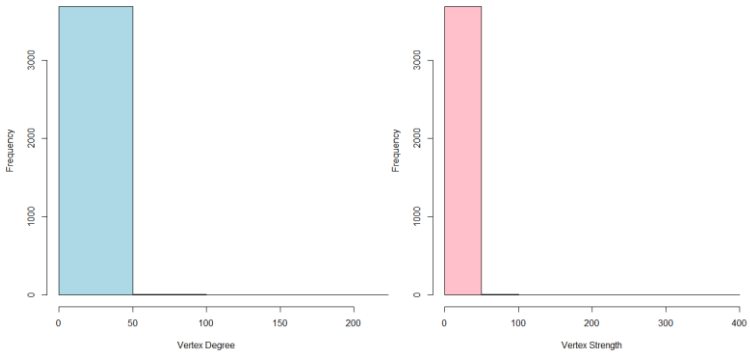

Fig. 53. Degree Distribution (left) and Strength (right) of PairToPort Network I

The average neighbor degree provides insight into the way in which vertices with different degrees are linked to each other. The plots corresponding to the five network types are again similar (Figures 54, 55, 56, 57, 58), suggesting that there is predisposition for vertices having low degrees to connect to vertices having rather higher degrees. It can be also noticed, that a high preponderance of nodes having the degree 1 link to nodes of both lower and higher degree. This confirms the behavior "one-to-several:browsing", which is described in Section 3.9.2.

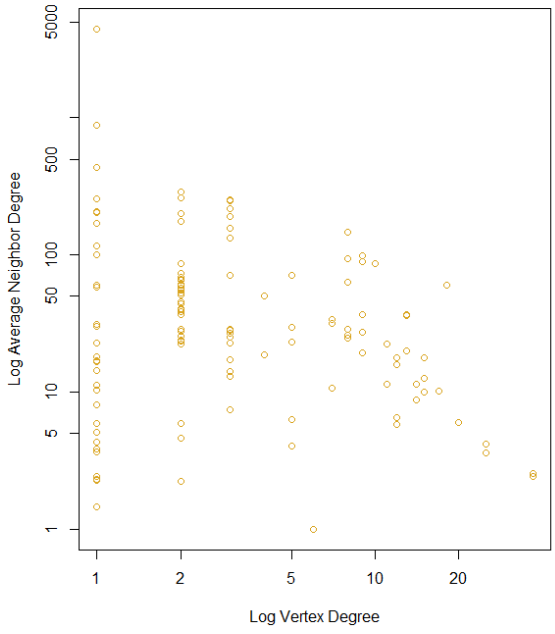

Fig. 54. Average Neighbor Degree versus Vertex Degree of the IP Network I

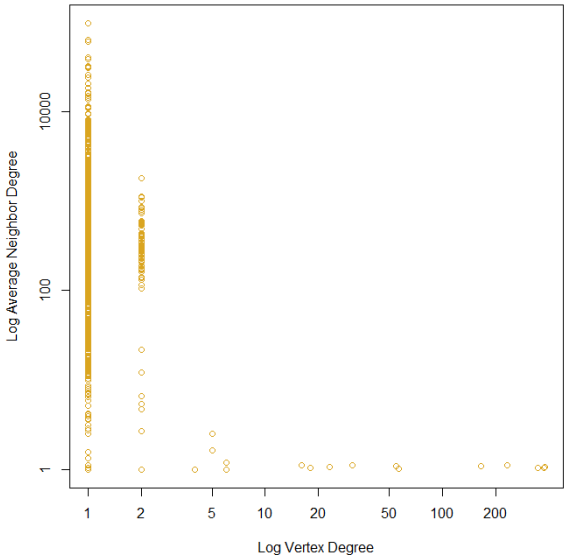

Fig. 55. Average Neighbor Degree versus Vertex Degree of the Port Network I

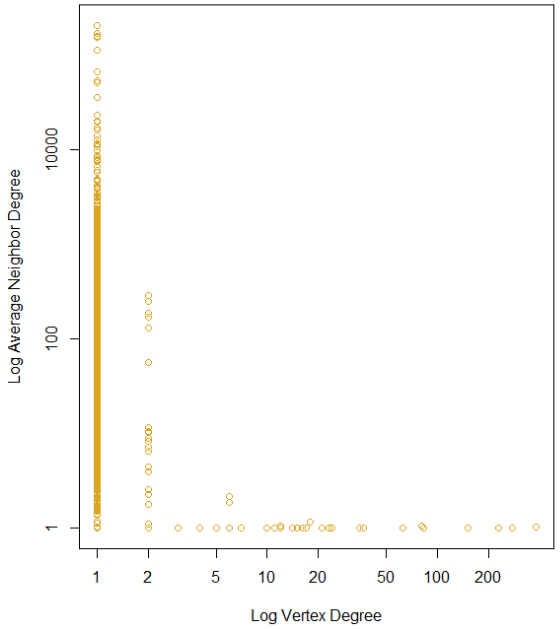

Fig. 56. Average Neighbor Degree versus Vertex Degree of the Pair Network I

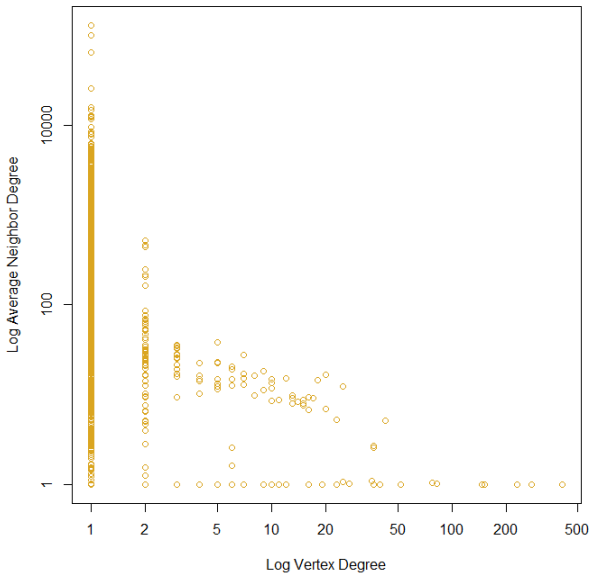

Fig. 57. Average Neighbor Degree versus Vertex Degree of the PairToIP Network I

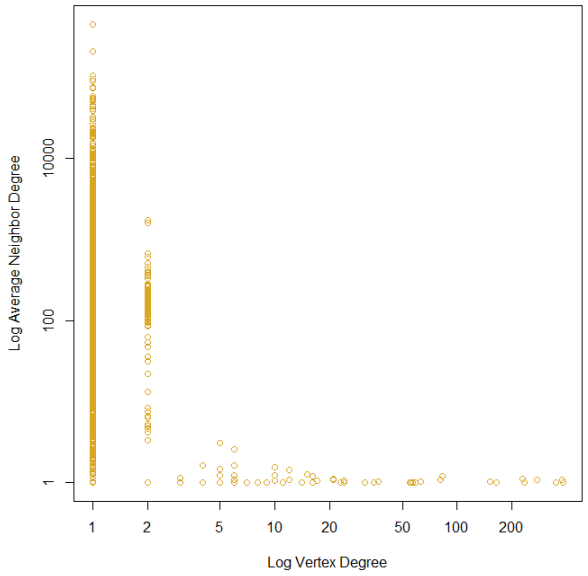

Fig. 58. Average Neighbor Degree versus Vertex Degree of the PairToPort Network I

Analyzing the connectivity of a graph is crucial in finding components that are not linked to each other. A highly disconnected graph, respectively network, can generate undesired consequences, since it is hardly manageable and is predisposed to attacks.

Thus, it is analyzed if the graph can be separated into distinct subgraphs and if not, the closeness to which it can be separated is quantified. On that account, it is determined how many edges or vertex sets need to be eliminated so that the graph becomes disconnected. An articulation point is a node whose removal disconnects the graph [30, p.58]. Associated to this issue is the question regarding the information flow within a network and finding the articulation points can indicate where the network is vulnerable.

A graph is connected if every node can be reached by any other node (i.e., if there is a walk between any two nodes) and a connected component is "a maximally connected subgraph" [30, p.69]. In random graph theory, a giant component is a connected component that dominates the others in magnitude [30, p.57].

All connected components within each graph that are assigned to one of the 5 graph types are enumerated in Figures 59, 61, 62, 63 and 64. It is visible that the network is disconnected, since the IP Network contains two components, and that there is a giant component, which contains 120 of the total 127 vertices of the graph (Figure 59). There are 15 articulation points (cut vertices) within the giant component, which corresponds to roughly 13% of the total vertices. A highly secure network aims to decrease this percentage as much as possible, so it is indicated to monitor the hosts whose labels are enumerated in Figure 60.

```
> comps <- decompose.graph(net_IP)
> table(sapply(comps, vcount))

  7 120
  1   1
```

Fig. 59. Connected Components of IP Network I

```
> net_IP.gc <- decompose.graph(net_IP)[[1]]
> net_IP.cut.vertices <- articulation.points(net_IP.gc)
> net_IP.cut.vertices
+ 15/120 vertices, named, from e750826:
 [1] 50 80 55 51 47 52 46 44 39 65 17 29 31 33 36
> length(net_IP.cut.vertices)
[1] 15
```

Fig. 60. Cut Vertices of IP Network I

The Port Network yields also a giant connected component, as illustrated in Figure 61. This component contains 1651 vertices, meaning that the endpoints of communication also tend to be interconnected.

```
> comps <- decompose.graph(net_Port)
> table(sapply(comps, vcount))

   1    2    3    5    7 1651
   4   14    4    1    3    1
```

Fig. 61. Connected Components of Port Network I

The giant component of the Pair Network contains 981 vertices (Figure 62). This network also includes several other components and it can be noticed that 40 components contain each 2 vertices, thus there are 40 edge records (bidirectional flows) being disconnected from the rest of the Pair Network from the perspective of (IP, port) heuristics. This means there are various interconnected (IP, port) pairs, which are detached from the rest of the Pair Network.

```
> comps <- decompose.graph(net_IP_Pair)
> table(sapply(comps, vcount))

  2   3   4   5   6   7   8  11  12  13  15  16  17  18  21  22  24  25  36  38  64  84 153 981
 40  12   9   5   7   4   5   3   1   1   1   1   1   1   1   2   1   2   1   3   1   1   1   1
```

Fig. 62. Connected Components of Pair Network I

The PairToIP Network (Figure 63) yields a giant component of 1164 vertices, meaning that the majority of the (IP, port) pairs tend to connect to the same IP, respectively that the majority of the IPs tend to connect to the same (IP, port) pair.

```
> comps <- decompose.graph(net_PairToIP)
> table(sapply(comps, vcount))

   2    3    4    5    6    7    8   10   11   12   13   17   20   24   38   41   53  148  153  169 1164
  10   11    6    5    3    2    2    1    3    1    3    1    1    1    2    1    1    1    1    1    1
```

Fig. 63. Connected Components of PairToIP Network I

The PairToPort Network (Figure 64) has two giant components of 1491, respectively 1511 vertices, meaning that the network tends to split in two parts, where each part yields the following characteristic: the (IP, port) pairs tend to connect to the same port, respectively the ports tend to connect to the same (IP, port) pair. This finding is consistent with the graph visualizations outlined in Figures 36, 37, 38 and 39.

```
> comps <- decompose.graph(net_PairPort)
> table(sapply(comps, vcount))

   2    3    4    5    6    7    8    9   10   12   15   17   24   32   36   56   57   58   60 1491 1511
  41   15    9    6    8    6    4    1    1    1    1    1    1    1    1    1    1    1    1    1    1
```

Fig. 64. Connected Components of PairToPort Network I

Graph partitioning can be seen as a tool for uncovering relevant subsets of the main network in the absence of knowing particular network features. Recall the fast-greedy.community method from Section 3.9.3, which uses an agglomerative hierarchical clustering algorithm to extract the number of communities and their size and to generate a hierarchy of embedded partitions of the graph. This algorithm is based on modularity, which stands for a set of nodes having a “higher intra-community edge density than a random graph” [37, p.1]. The Figures 65, 66, 67, 68 and 69 highlight the community size for each of the 5 network dimensions. It is visible that only the IP Network has a community that dominates in size all the other communities (46 vertices) and has the lowest number of communities (8 communities).

```
> length(kc)
[1] 8
> sizes(kc)
Community sizes
 1  2  3  4  5  6  7  8
16 15  8 11 46 17  7  7
```

Fig. 65. Community Size IP Network I

A few communities of the Port Network (Figure 66), Pair Network (Figure 67) and PairToIP Network (Figure 68) dominate in size all the other communities, which leads to the higher number of total communities when compared to the IP Network.

```
> length(kc)
[1] 37
> sizes(kc)
Community sizes
  1   2   3   4   5   6   7   8   9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28  29  30  31  32  33  34  35  36  37
353 359 333 228  38 166  58  56  24  19  17   7   7   5   3   3   3   3   2   2   2   2   2   2   2   2   2   2   2   2   2   7   1   1   1   1
```

Fig. 66. Community Size Port Network I

```
> length(kc)
[1] 65
> sizes(kc)
Community sizes
  1   2   3   4   5   6   7   8   9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28  29  30  31  32  33  34  35  36
277 230 407 115  83  78 153 148  61  28  53  41  54  38  38  24  20  17  13  13  13  12  11  11  11  10   8   8   7   7   6   6   6   5   5   5
 37  38  39  40  41  42  43  44  45  46  47  48  49  50  51  52  53  54  55  56  57  58  59  60  61  62  63  64  65
  5   5   4   4   4   4   4   4   3   3   3   3   3   3   3   3   3   3   2   2   2   2   2   2   2   2   2   2
```

Fig. 67. Community Size Pair Network I

```
> length(kc)
[1] 109
> sizes(kc)
Community sizes
  1   2   3   4   5   6   7   8   9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28  29  30  31  32  33  34  35  36  37  38  39  40  41
381 276 230  81 153  84  64  38  38  38  36  25  25  13  24  21  22  22  18  17  16  15  13  12  11  11  11   8   8   8   8   8   7   7   7   6   6   6   6   6
 42  43  44  45  46  47  48  49  50  51  52  53  54  55  56  57  58  59  60  61  62  63  64  65  66  67  68  69  70  71  72  73  74  75  76  77  78  79  80  81  82
  6   6   5   5   5   5   5   4   4   4   4   4   4   4   4   4   3   3   3   3   3   3   3   3   3   3   3   3   2   2   2   2   2   2   2   2   2   2   2   2   2
 83  84  85  86  87  88  89  90  91  92  93  94  95  96  97  98  99 100 101 102 103 104 105 106 107 108 109
  2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2
```

Fig. 68. Community Size PairToIP Network I

The visual representations resulted by plotting the IP Network (Figure 69), the Pair Network (Figure 70) and the Port Network (Figure 71) support a good overview to understand how the communities are spread within the networks and how they connect to each other.

Based on what has already been identified with respect to the network, it is reasonable to suppose that the largest community of the IP Network (the fifth community – light blue color in Figure 70) is centered around the nodes 85 and 56, which together with their direct neighborhoods form the largest egocentric subgraphs (Figure 40, Figure 41).

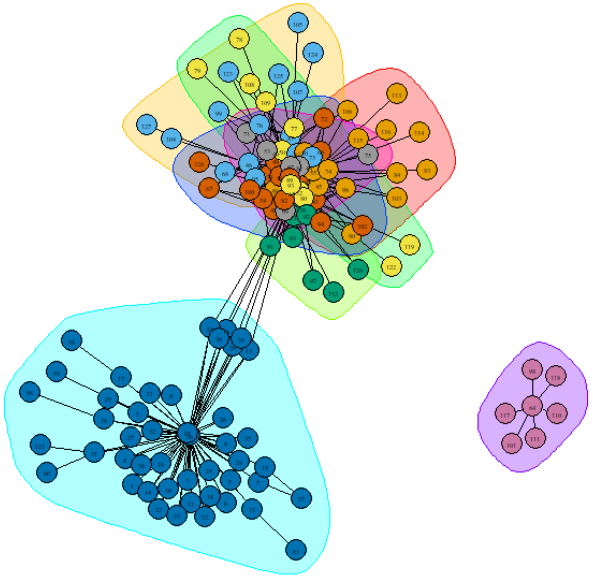

Fig. 69. Community Visualization IP Network I

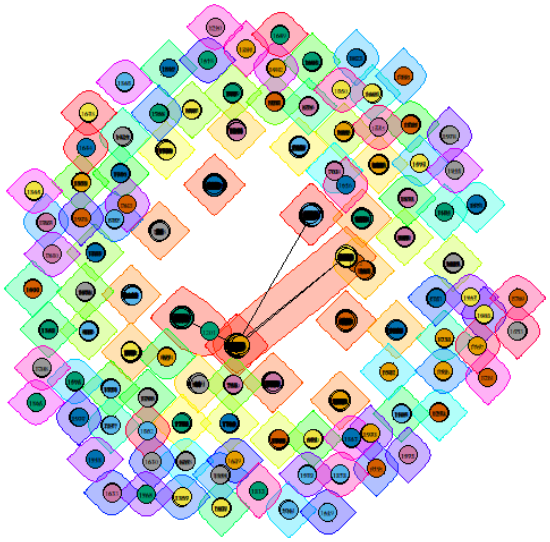

Fig. 70. Community Visualization Pair Network I

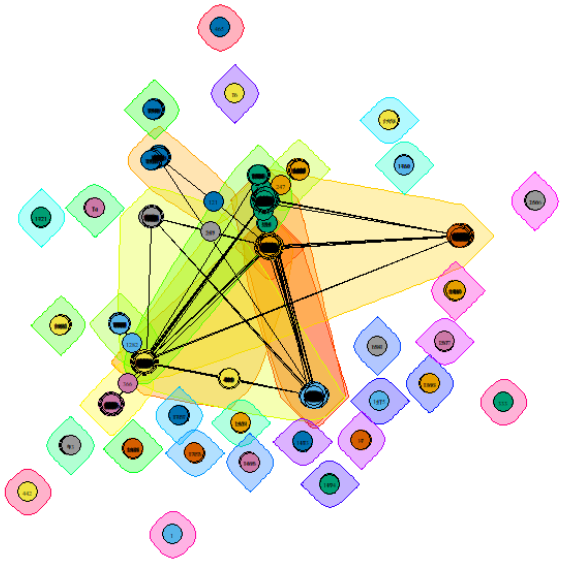

Fig. 71. Community Visualization Port Network I

The resulting hierarchy of the IP Network can be visualized in form of a dendrogram in Figure 72. The values on the right side of the figure illustrate the individual IP addresses. As we move to the left side of the tree, these nodes link together to create bigger and bigger communities, as signified by the edges. The horizontal width of the split nodes in the tree signifies the series in which the splits occur. On the left side of the tree, all IP addresses are linked together in an individual community. It is assumed, that once a case is merged into a cluster, it will not be classified again [13, p.213]. A cross section of the dendrogram at any level illustrates the communities at that level [24, p.2].

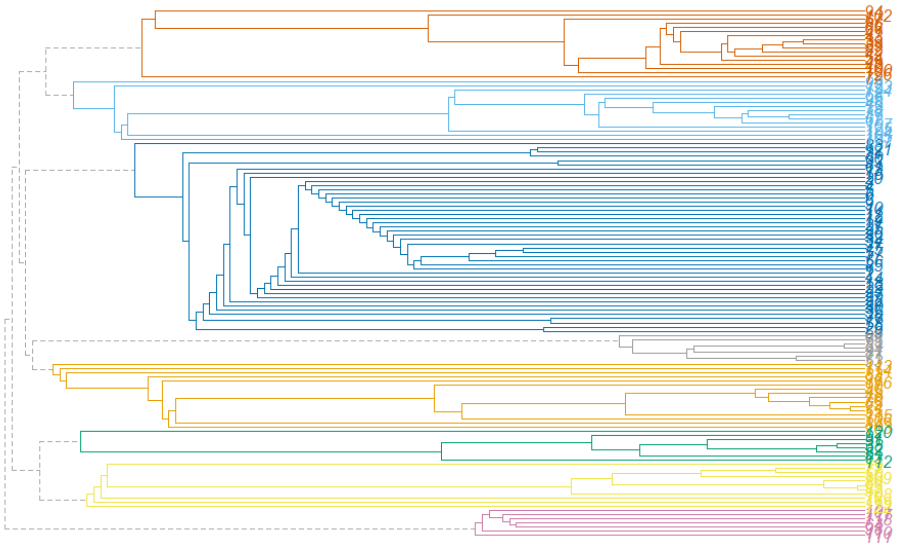

Fig. 72. Dendrogram of IP Network I

Since it is expected that the emergence of cohesive fractions of vertices as a result of graph partitioning leads to a similarity in specific relevant attributes of the network, each attribute class is compared and contrasted against its community affiliation. For this purpose, three two-dimensional tables show the grouping of each IP (Figure 73), port (Figure 74) and pair (Figure 75) according to their membership under each classification. As expected, the fifth community of the IP Network largely captures the classes of hosts being assigned to none of the common services, which also form the two largest egocentric subgraphs (Figure 40, Figure 41). As for the Port Network (Figure 74) and Pair Network (Figure 75), the nodes are spread over all functional classes, which makes their assigned communities less explainable against the validation sets.

	DNSClient	DNSHost	DNSServer	webClient	webHost	webServer	remoteHost	mailHost	invalid_Tos	VPN	/Worm.Opasoft, W32.Opaserv.Worm	NONP2P	P2P	P2PNONP2P
1	3	12	1	2	14	0	16	16	0	3	4	11	1	4
2	2	13	0	1	11	3	15	15	0	2	0	7	3	5
3	2	6	0	2	4	2	8	8	3	2	0	4	1	3
4	3	8	0	3	6	2	11	11	2	2	0	3	4	4
5	0	46	0	0	46	0	46	46	0	0	0	2	7	37
6	5	12	0	7	9	1	17	17	1	8	0	6	2	9
7	0	7	0	0	7	0	7	7	0	0	0	0	7	0
8	0	7	0	2	5	0	7	7	0	2	0	1	2	4

Fig. 73. Functional Classes IP Network I

	func.class			
.m	NONP2P	P2P	P2PNONP2P	
1	353	0		0
2	359	0		0
3	333	0		0
4	228	0		0
5	31	6		1
6	81	77		8
7	58	0		0
8	56	0		0
9	0	23		1
10	0	18		1
11	0	14		3
12	0	7		0
13	0	7		0
14	0	5		0
15	0	3		0
16	0	3		0
17	0	3		0
18	0	3		0
19	0	2		0
20	0	2		0
21	0	2		0
22	0	2		0
23	0	2		0
24	0	2		0
25	0	2		0
26	2	0		0
27	0	2		0
28	0	2		0
29	0	2		0
30	0	2		0
31	0	2		0
32	0	2		0
33	0	7		0
34	1	0		0
35	1	0		0
36	1	0		0
37	1	0		0

Fig. 74. Functional Classes Port Network I

	DNSClient	DNSHost	DNSServer	webClient	webHost	webServer	remoteHost	mailHost	invalid_Tos	VPN	/Worm.Opasoft, W32.Opaserv.Worm	NONP2P	P2P
1	362	18	1	346	35	0	381	381	0	0	0	381	0
2	206	70	0	252	23	1	276	276	0	0	0	276	0
3	204	26	0	221	8	1	230	230	76	0	0	230	0
4	2	79	0	2	79	0	81	81	0	0	0	81	0
5	152	1	0	152	0	1	153	153	104	0	0	153	0
6	9	75	0	83	1	0	84	84	0	0	0	84	0
7	63	1	0	62	2	0	64	64	0	0	0	64	0
8	0	38	0	0	38	0	38	38	0	0	0	38	0
9	37	1	0	37	1	0	38	38	0	0	0	38	0
10	0	38	0	0	38	0	38	38	0	0	0	0	38
11	35	1	0	35	0	1	36	36	0	0	0	36	0
12	24	1	0	22	3	0	25	25	0	0	0	25	0
13	0	25	0	0	25	0	25	25	0	0	0	0	25
14	12	1	0	12	1	0	13	13	0	0	0	13	0
15	0	24	0	0	24	0	24	24	0	0	0	0	24
16	14	7	0	15	6	0	21	21	0	19	0	21	0
17	15	6	1	10	12	0	22	22	0	0	0	22	0
18	17	5	0	19	3	0	22	22	0	0	0	22	0
19	0	18	0	0	18	0	18	18	0	0	0	0	18
20	0	17	0	0	17	0	17	17	0	0	0	0	17

Fig. 75. Functional Classes Pair Network I

In order to see if a parallel can be made between degree, betweenness, total transfer, neighbors, duration and closeness, these measures are compared between each

other within each network type (Figure 76, Figure 77, Figure 78 and Figure 79) by using the Pearson correlation coefficient[25]. This approach facilitates a better comprehension of the network behavior in terms of the "social positions" of its actors. The results show a high correlation between betweenness and degree over all network types, which indicates that the hosts that send a lot of information are also between other pair of hosts, respectively they have many shortest paths passing through them. There is also a high correlation between degree and total transfer in bytes, indicating that the more links a given vertex has to other vertices within the network, the more bytes are being transferred. This also signifies that there is not a large amount of bytes being sent on a single link, but this amount is spread along multiple links. As expected, there is also a high correlation between degree and neighbors, since the degree measures all the ingoing and outgoing links, while the neighbors are the distinct vertices being linked to a particular vertex. High correlation values between degree and neighbors indicate that the number of links between two adjacent vertices is small. A correlation of 1 can be achieved when there is only one interaction between two adjacent hosts. The low values of the correlation between closeness and all the other variables is the result of applying the closeness measure on disconnected networks, which does not provide an accurate estimation. The correlation between the duration in seconds and the degree is below 50%, which indicates that a higher number of links between two adjacent vertices does not signify a higher duration of the transfer, but it may also be the case that vertices connected by a few links send over higher amounts of time.

[25] https://en.wikipedia.org/wiki/Pearson_correlation_coefficient

```
> cor(nodes_IP$Betweenness,nodes_IP$Degree)
[1] 0.7234464
> cor(nodes_IP$Betweenness,nodes_IP$Community)
[1] -0.1469364
> cor(nodes_IP$Betweenness,nodes_IP$TotalTransfer)
[1] 0.5418085
> cor(nodes_IP$Degree,nodes_IP$Community)
[1] -0.1468854
> cor(nodes_IP$Degree,nodes_IP$TotalTransfer)
[1] 0.9343288
> cor(nodes_IP$Neighbors,nodes_IP$TotalTransfer)
[1] 0.561227
> cor(nodes_IP$Neighbors,nodes_IP$Betweenness)
[1] 0.6707888
> cor(nodes_IP$Neighbors,nodes_IP$Community)
[1] -0.01604704
> cor(nodes_IP$Neighbors,nodes_IP$Degree)
[1] 0.6237587
> cor(nodes_IP$Closeness,nodes_IP$TotalTransfer)
[1] 0.1833726
> cor(nodes_IP$Closeness,nodes_IP$Betweenness)
[1] 0.1807914
> cor(nodes_IP$Closeness,nodes_IP$Community)
[1] -0.3179779
> cor(nodes_IP$Closeness,nodes_IP$Degree)
[1] 0.198151
> cor(nodes_IP$Closeness,nodes_IP$Neighbors)
[1] 0.2698624
> cor(nodes_IP$Duration,nodes_IP$TotalTransfer)
[1] 0.4393652
> cor(nodes_IP$Duration,nodes_IP$Betweenness)
[1] 0.1849194
> cor(nodes_IP$Duration,nodes_IP$Community)
[1] 0.1371162
> cor(nodes_IP$Duration,nodes_IP$Degree)
[1] 0.4941349
> cor(nodes_IP$Duration,nodes_IP$Neighbors)
[1] 0.4369743
>
```

Fig. 76. Correlation of Continuous Variables IP Network I

```
> cor(nodes_port$Betweenness,nodes_port$Degree)
[1] 0.9866753
> cor(nodes_port$Betweenness,nodes_port$Community)
[1] -0.006991145
> cor(nodes_port$Betweenness,nodes_port$TotalTransfer)
[1] 0.91545
> cor(nodes_port$Degree,nodes_port$Community)
[1] -0.006313287
> cor(nodes_port$Degree,nodes_port$TotalTransfer)
[1] 0.9333423
> cor(nodes_port$Neighbors,nodes_port$TotalTransfer)
[1] 0.9356896
> cor(nodes_port$Neighbors,nodes_port$Betweenness)
[1] 0.9866173
> cor(nodes_port$Neighbors,nodes_port$Community)
[1] -0.007187691
> cor(nodes_port$Neighbors,nodes_port$Degree)
[1] 0.9999494
> cor(nodes_port$Closeness,nodes_port$TotalTransfer)
[1] -0.005200912
> cor(nodes_port$Closeness,nodes_port$Betweenness)
[1] 0.02299746
> cor(nodes_port$Closeness,nodes_port$Community)
[1] -0.8120068
> cor(nodes_port$Closeness,nodes_port$Degree)
[1] 0.01097468
> cor(nodes_port$Closeness,nodes_port$Neighbors)
[1] 0.01221106
> cor(nodes_port$Duration,nodes_port$TotalTransfer)
[1] 0.662225
> cor(nodes_port$Duration,nodes_port$Betweenness)
[1] 0.5480077
> cor(nodes_port$Duration,nodes_port$Community)
[1] 0.06378487
> cor(nodes_port$Duration,nodes_port$Degree)
[1] 0.568642
> cor(nodes_port$Duration,nodes_port$Neighbors)
[1] 0.5690682
```

Fig. 77. Correlation of Continuous Variables Port Network I

```
> cor(nodes_Pair$Betweenness,nodes_Pair$Degree)
[1] 0.8306091
> cor(nodes_Pair$Betweenness,nodes_Pair$Community)
[1] -0.02859889
> cor(nodes_Pair$Betweenness,nodes_Pair$TotalTransfer)
[1] 0.6665844
> cor(nodes_Pair$Degree,nodes_Pair$Community)
[1] -0.01691796
> cor(nodes_Pair$Degree,nodes_Pair$TotalTransfer)
[1] 0.9016497
> cor(nodes_Pair$Neighbors,nodes_Pair$TotalTransfer)
[1] 0.9016221
> cor(nodes_Pair$Neighbors,nodes_Pair$Betweenness)
[1] 0.8306573
> cor(nodes_Pair$Neighbors,nodes_Pair$Community)
[1] -0.01692888
> cor(nodes_Pair$Neighbors,nodes_Pair$Degree)
[1] 0.9999969
> cor(nodes_Pair$Closeness,nodes_Pair$TotalTransfer)
[1] -0.02750523
> cor(nodes_Pair$Closeness,nodes_Pair$Betweenness)
[1] 0.05370712
> cor(nodes_Pair$Closeness,nodes_Pair$Community)
[1] -0.5519595
> cor(nodes_Pair$Closeness,nodes_Pair$Degree)
[1] 0.007927758
> cor(nodes_Pair$Closeness,nodes_Pair$Neighbors)
[1] 0.008008973
> cor(nodes_Pair$Duration,nodes_Pair$TotalTransfer)
[1] 0.5714832
> cor(nodes_Pair$Duration,nodes_Pair$Betweenness)
[1] 0.4103242
> cor(nodes_Pair$Duration,nodes_Pair$Community)
[1] 0.002711888
> cor(nodes_Pair$Duration,nodes_Pair$Degree)
[1] 0.5617021
> cor(nodes_Pair$Duration,nodes_Pair$Neighbors)
[1] 0.5616255
```

Fig. 78. Correlation of Continuous Variables IPPair Network I

```
> cor(nodes_PairToIP$Betweenness,nodes_PairToIP$Degree)
[1] 0.8092682
> cor(nodes_PairToIP$Betweenness,nodes_PairToIP$Community)
[1] -0.03021237
> cor(nodes_PairToIP$Degree,nodes_PairToIP$Community)
[1] -0.01462956
> cor(nodes_PairToIP$Neighbors,nodes_PairToIP$Betweenness)
[1] 0.8092682
> cor(nodes_PairToIP$Neighbors,nodes_PairToIP$Community)
[1] -0.01462956
> cor(nodes_PairToIP$Neighbors,nodes_PairToIP$Degree)
[1] 1
> cor(nodes_PairToIP$Closeness,nodes_PairToIP$Betweenness)
[1] 0.05505032
> cor(nodes_PairToIP$Closeness,nodes_PairToIP$Community)
[1] -0.6029515
> cor(nodes_PairToIP$Closeness,nodes_PairToIP$Degree)
[1] -0.01242685
> cor(nodes_PairToIP$Closeness,nodes_PairToIP$Neighbors)
[1] -0.01242685
```

Fig. 79. Correlation of Continuous Variables PairToIP Network I

A centrality plot for each of the network types (Figure 80, Figure 81, Figure 82, Figure 83 and Figure 84) illustrates in parallel the measures of degree, closeness and betweenness (in this order from left to right) and it is very useful to identify outliers or similar behavior within the network behavior. Similar to the findings provided by the degree histogram, it is visible in the centrality plots of all network types, that most of the nodes have rather lower degrees, while 3 nodes have a degree definitely higher

than the average The closeness distribution along all nodes is irregular, although one single node has a higher closeness than the average closeness. This particular node also has the highest betweenness and is among the nodes having a high degree. The betweenness centrality has a similar pattern with degree: many nodes with a low betweenness and a few nodes with a high betweenness. The nodes having a high betweenness are also the nodes having a high degree. The hosts having high values of centrality values highlight the way traffic flows, by suggesting the speed of information transfer and receiving frequency.

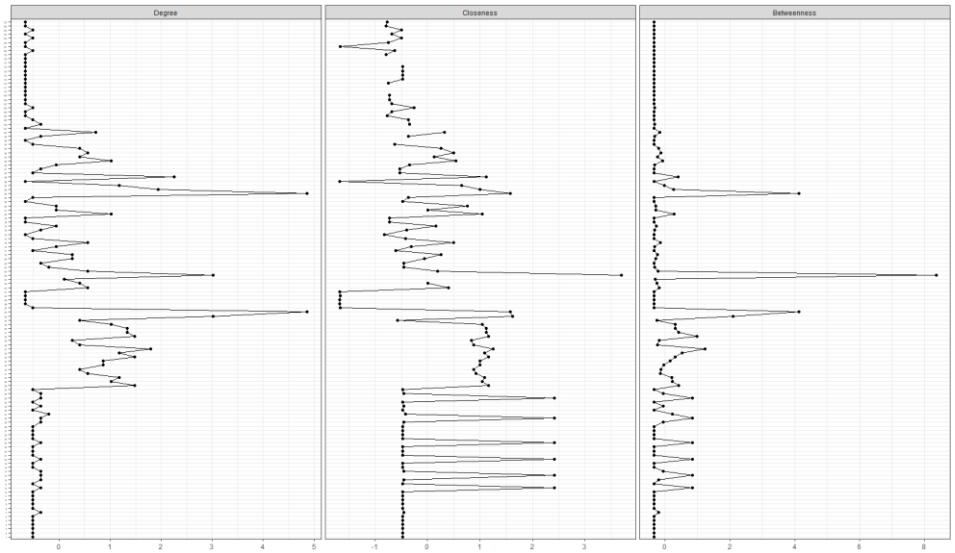

Fig. 80. Centrality Plot IP Network I

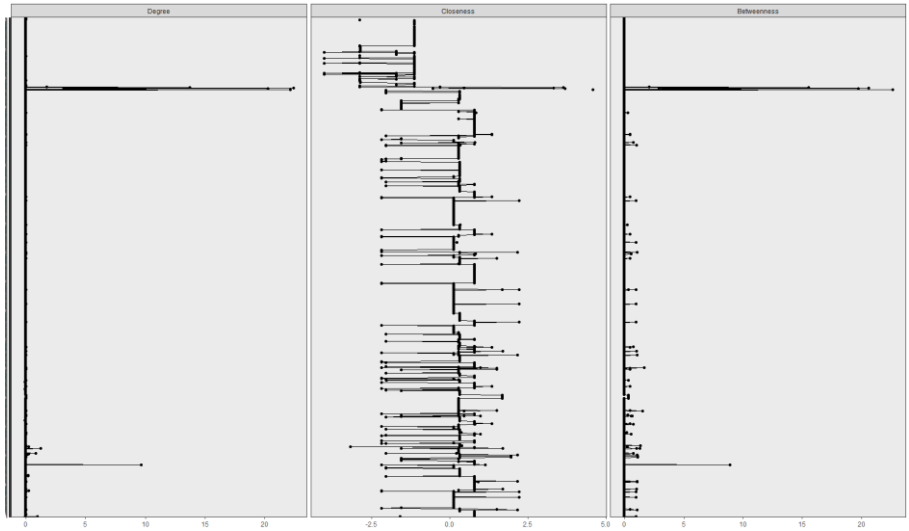

Fig. 81. Centrality Plot Port Network I

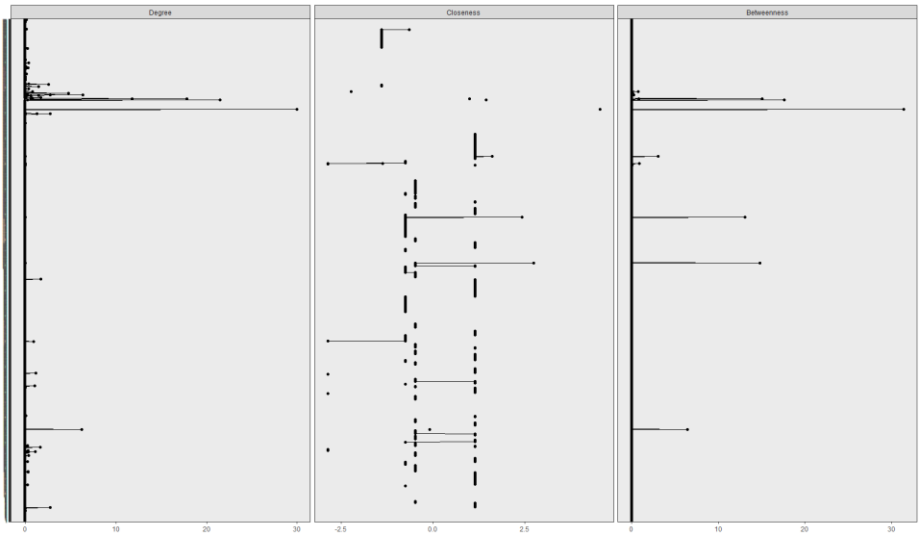

Fig. 82. Centrality Plot Pair Network I

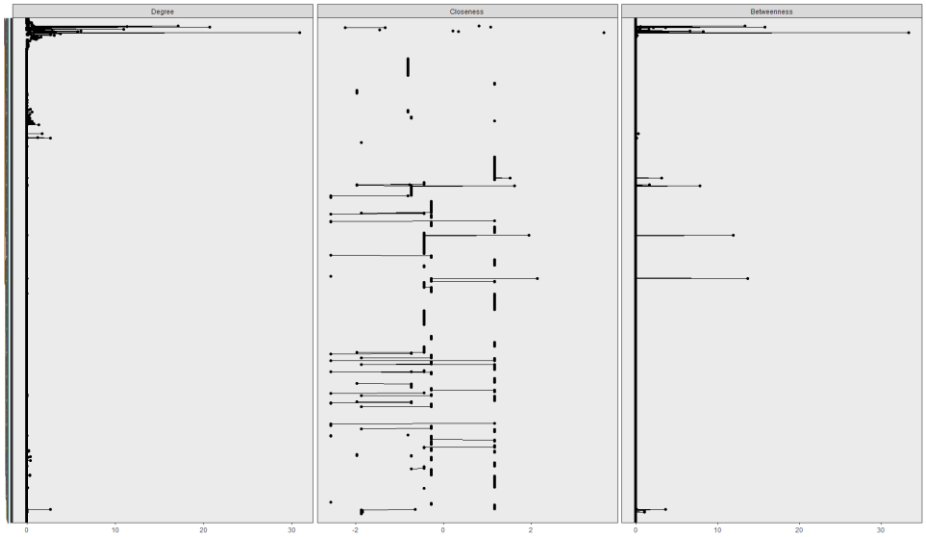

Fig. 83. Centrality Plot PairToIP Network I

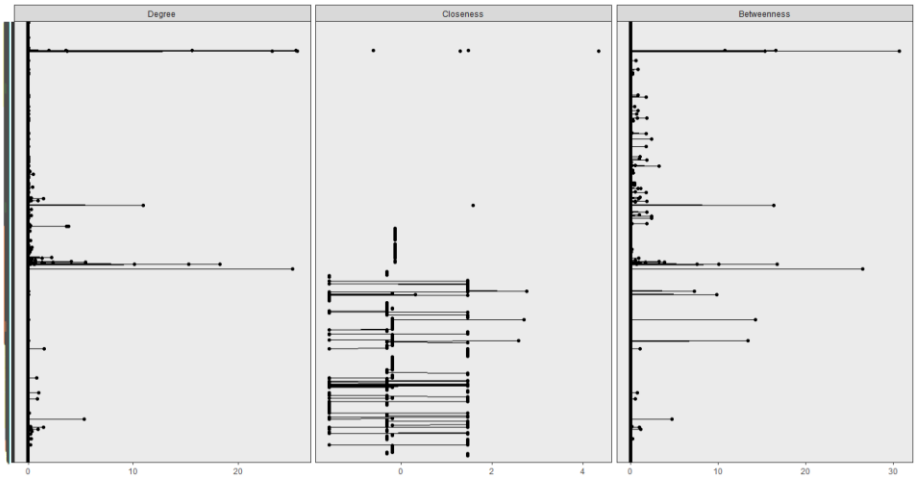

Fig. 84. Centrality Plot PairToPort Network I

Recall that the structural equivalence of the network is defined by the pair of actors where each of the actors has the same relationships to a particular set of actors. In the first phase of the analysis, the exactly structural equivalent hosts are defined, by creating the adjacency matrix of the network and indexing the duplicated rows. Since it is relevant to analyze the distinct connections of a host (neighbors) and not its degree, as resulted from the initial adjacency matrix, the non-zero values of the adjacency matrix

are replaced by 1. The resulted indexes of the duplicated rows define the IDs of the structural equivalent nodes and the column sums for each row support in finding the corresponding pair of nodes. It is relevant to identify structural equivalent pair of hosts that have many neighbors, since low orders of magnitude regarding neighbors are considered "normal" within a network traffic sample, while high orders of magnitude may indicate a suspicious activity that needs to be investigated. For this purpose, all hosts in a structural equivalent pair that have more than 10 neighbors are considered. Figure 85 shows the result of applying this methodology to the IP Network.

NodeLabel	SumCol	IP
56	37	146.16.251.99
85	37	146.16.250.41

Fig. 85. Exactly Structural Equivalent Hosts of IP Network I

The result in Figure 85 highlights the two hosts responsible also for the highest egocentric subnetworks (as outlined in Fig.40 and Fig 41). There are two possibilities here: either there are two distinct hosts that coincidentally have the same number of neighbors, or there is only one machine/virtual host, which has two network interface cards[26].

In the second phase, approximately equivalent set of hosts are analyzed, by identifying the structural equivalent blocks or partitions. This approach describes the character of each role regarding its expected pattern of connections with other roles [13, p.250]. Figure 86 describes the block membership and the density of each block within the matrix that corresponds to the network.

```
Network Blockmodel:

Block membership:

  1   2   3   4   5   6   7   8   9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27
  1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1
 28  29  30  31  32  33  34  35  36  37  38  39  40  41  42  43  44  45  46  47  48  49  50  51  52  53  54
  1   1   1   1   1   1   1   1   1   1   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2
 55  56  57  58  59  60  61  62  63  64  65  66  67  68  69  70  71  72  73  74  75  76  77  78  79  80  81
  2   1   2   3   3   3   3   4   3   1   4   4   4   3   4   4   3   4   4   4   4   3   4   3   3   2   4
 82  83  84  85  86  87  88  89  90  91  92  93  94  95  96  97  98  99 100 101 102 103 104 105 106 107 108
  2   4   4   1   4   4   3   4   4   3   3   4   4   4   4   3   3   3   4   3   4   4   3   3   4   3   3
109 110 111 112 113 114 115 116 117 118 119 120 121 122 123 124 125 126 127
  3   3   3   3   4   4   4   4   3   3   4   4   3   4   3   3   3   4   3

Reduced form blockmodel:

         1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32 33 34 35 36
37 38 39 40 41 42 43 44 45 46 47 48 49 50 51 52 53 54 55 56 57 58 59 60 61 62 63 64 65 66 67 68 69 70 71 72
73 74 75 76 77 78 79 80 81 82 83 84 85 86 87 88 89 90 91 92 93 94 95 96 97 98 99 100 101 102 103 104 105 106
 107 108 109 110 111 112 113 114 115 116 117 118 119 120 121 122 123 124 125 126 127
              Block 1     Block 2   Block 3     Block 4
Block 1 0.094871795 0.002380952 0.0093750 0.004411765
Block 2 0.002380952 0.000000000 0.0639881 0.281512605
Block 3 0.009375000 0.063988095 0.0000000 0.000000000
Block 4 0.004411765 0.281512605 0.0000000 0.000000000
```

Fig. 86. Equivalent Blocks of IP Network I

[26] https://en.wikipedia.org/wiki/Network_interface_controller

The representation of the block model in Figure 87 shows that most of the comparably dense blocks are outside the diagonal. These are marginal roles (blocks 2, 3 and 4) that are connected to the center (block 1).

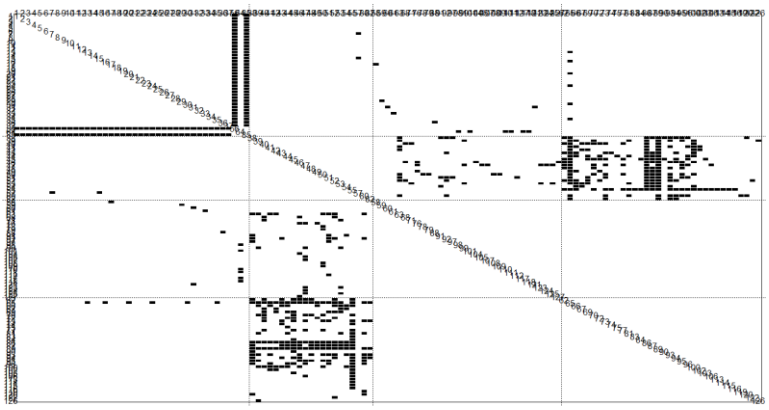

Fig. 87. Representation of Equivalent Blocks IP Network I

The blocks can be visualized as a graph in Fig. 88 and it can be noticed that the clique containing the two exactly structural equivalent hosts (56 and 85) represents the core block, labeled with the color orange.

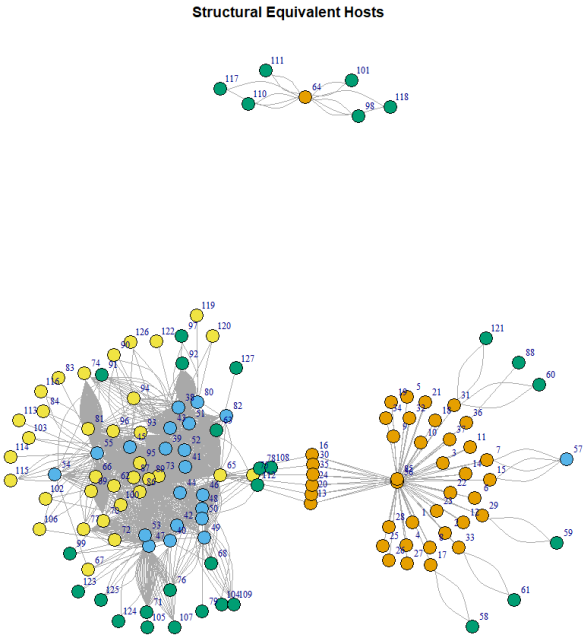

Fig. 88. Visualization of the Structural Equivalence Blocks of Network I

The time plots over the 4 dimensions: transfer in bytes, degree, betweeenness and neighbors reveal the critical time intervals but also the irregularity within the network behavior. It is visible in Figure 89 that the highest amount of bytes are transferred at the beginning (minute 10:59) of the data log – around 2187 bytes. On the contrary, the rest of the time interval contains considerably low bytes transfers.

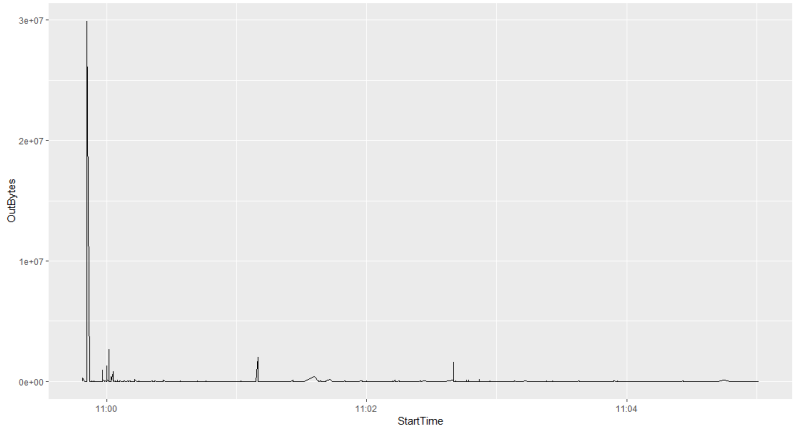

Fig. 89. Bytes Transfer versus Time IP Network I

The degree distribution over time is characterized by a roughly constant dispersion in the first half of the time interval, followed by a rather irregular and high-value dispersion in the second half of the interval (Figure 90).

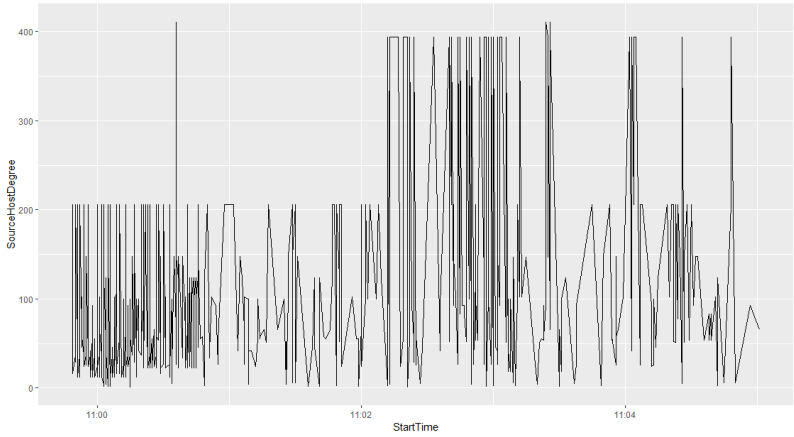

Fig. 90. Degree versus Time IP Network I

The neighbors distribution over time indicates a constant and high-value dispersion in the first minute of the time interval, followed by a rather regular dispersion in rest

of the interval (Figure 91). By analyzing the degree and the neighbors distribution over time following conclusion can be drawn: between the time interval 11:02:15 and 11:03:30 the ratio of degree:neighbors is roughly 16, indicating the existence of hosts that have many links to destinations but at the same time target a few distinct destinations. On the contrary, the ration degree:neighbors in the first minute of the total time interval is roughly 4.

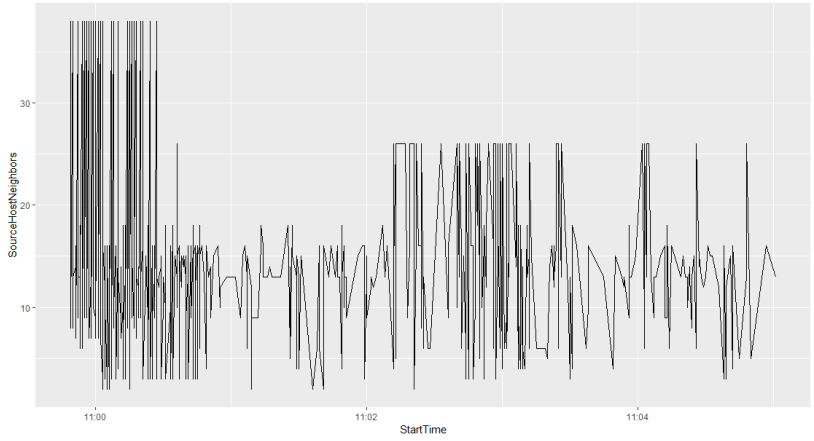

Fig. 91. Neighbors versus Time IP Network I

The betweenness distribution over time (Figure 92) indicates a similar behavior with degree, which is to be expected according to the Pearson correlation of 0,72 between degree and betweenness (Figure 76). The three peaks in Figure 92 are conform with the centrality plot of the IP Network in Figure 80, meaning that only 3 hosts are responsible for the 3 highest betweenness values over time.

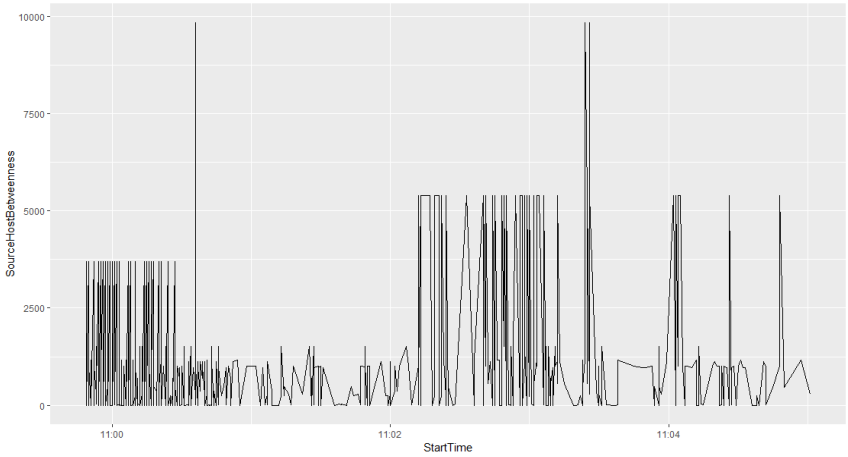

Fig. 92. Betweenness versus Time IP Network I

All measures outlined above and their outcomes result in 208 flow attributes mapped in the main data frame. The correlation results provide an indication on where to look further for possible anomalies/threads, by monitoring the hosts serving as basis for the outliers. Since the P2P algorithm outlined in Section 3.7 is complex and demands many steps for identifying the P2P flows, it is useful to determine if the results of the SNA can be used to detect the P2P traffic by means of a lower computational effort. The question here is if it is possible to replace the P2P algorithm illustrated in Figure 11 by a set of SNA heuristics.

A positive value of the difference between the neighbors of the PairToIP network corresponding to the source pair and the neighbors of the PairToPort network corresponding to the destination pair results in 404 flows, from which 283 are identified as P2P by the P2P algorithm in Section 3.7. Restraining the selection to high ports (both source and destination port > 1024) and to the TCP and UDP protocols results in a number of 318 flows, from which 283 are P2P flows (according to the P2P algorithm in Section 3.7). Nevertheless, the P2P algorithm in Section 3.7 identified a total number of 285 flows, so two P2P flows are missing from the applied SNA method. This method consisting of 3 steps filters with an accuracy of approximately 89% all P2P flows within the network.

Recall that P2P is not secure, so it useful to monitor all assets participating in P2P traffic. By sorting the ports engaging in P2P traffic by their total transfer, the ports sending the highest amount of bytes are 3268 and 5061. A closer examination of these ports leads to the conclusion that both ports are not secure against active or passive attacks. Port 3268 is used by LDAP[27] (Lightweight Directory Access Protocol), where the data is transferred without being encrypted and is predisposed to injected requests and manipulated streams. Port 5061 is used by SIP[28](session initiation protocol) and is exposed to attacks when left unprotected. The next step for the user would be to examine the hosts assigned to these ports.

5 Conceptual Propositions

The findings from Section 4 lead to the formulation of the following conceptual propositions that are further validated with the second data set:

1. The graph corresponding to the network traffic within an enterprise company is usually disconnected, which means not all devices are interconnected.
2. At least one small component of this disconnected graph has exclusively P2P traffic.
3. HTTP(S) is one of the most used services within an enterprise company.

[27] https://wiki.wireshark.org/LDAP

[28] https://en.wikipedia.org/wiki/Session_Initiation_Protocol

4. Servers tend to be placed within one single network component yielding a high edge density or within one network cluster, independently from their prefix assignment.
5. The egocentric subgraphs with the largest amount of neighbors contain at least one server.
6. According to degree distribution and graph strength, most of the hosts have up to 50 links, independently of the time frame.
7. The hosts of low degrees connect to hosts of higher degrees.
8. The general behavior type of the network is “one-to-many:browsing”, except there are anomalies occurring within the traffic.
9. There is at least one articulation point within a connected network whose removal disconnects the network; all articulation points should be monitored closely.
10. The communities resulted from graph partitioning exhibit no correlation to the attribute sets (or functional classes).
11. Betweenness and degree are highly intercorrelated over all network types, which signifies that the hosts sending a lot of information are also between other pair of hosts, respectively they have many shortest paths passing through them.
12. The majority of the (IP, port) pairs tend to connect to the same IP, respectively that the majority of the IPs tend to connect to the same (IP, port) pair.
13. The network tends to split in two parts, where each part yields the following characteristic: the (IP, port) pairs tend to connect to the same port, respectively the ports tend to connect to the same (IP, port) pair.
14. Degree and total transfer in bytes are highly correlated, indicating that the more links a given vertex has to other vertices within the network, the more bytes are being transferred. This also signifies that bytes are not being sent in large amounts across one single link, but this amount is spread into chunks along multiple links;
15. Degree and neighbors are highly correlated, since the degree measures all the ingoing and outgoing links, while the neighbors are the distinct linked to a particular vertex. High correlation values between these variables indicate that the number of links between two adjacent vertices is small. A correlation of one can be achieved when there is only one link between two adjacent hosts.
16. The correlation between the duration in seconds and the degree is below 50%, which indicates that a higher number of links between two adjacent vertices does not signify a high duration of the transfer, but it may also be the case that vertices connected by a few links transfer packets over high amounts of time.
17. Most of the DNS clients are also web clients.

18. Most of the nodes have rather lower degrees, while a small fraction of nodes have a degree definitely higher than the average. A high degree may indicate the presence of suspicious activity.
19. The closeness distribution tends to be irregular along all nodes. Nodes having a high closeness have favored positions, since they can be reached by other nodes at shorter path lengths, thus they act as reference points for other nodes.
20. The betweenness centrality has a similar pattern with degree: many nodes with a low betweenness and a few nodes with a high betweenness. The nodes having a high betweenness are also the nodes having a high degree.
21. The core structural equivalence block contains at least one egocentric subnetwork.
22. The highest amount of bytes tend to be sent within one single time faction.
23. If the difference between the neighbors of the PairToIP network corresponding to the source pair and the neighbors of the PairToPort network corresponding to the destination pair is greater or equal to zero, the ports are above 1024 and the protocol is TCP/UDP, then more than 85% of the total P2P flows are identified.

These 23 propositions are evaluated in Section 6 by applying the same methodology from Section 4 on the LBNL data set.

6 Evaluation Based on Data Set II

The second data set belongs to LBNL/ICSI Enterprise Tracing Project and is publicly available on the homepage[29]. The data describes again an internal enterprise traffic and was recorded at a medium-sized site with the purpose of providing a resource for others to analyse patterns within enterprises[30]. LBNL collected packet traces spanning over 100 hours of traffic from thousands of hosts. The raw data file used for the evaluation belongs to the collection of data files provided by LBNL and contains 388417 anonymized flows and captures the traffic occurring in a period of approximately one hour. The file was loaded into the R environment and converted to a data frame, for the intent of further analysis. The column headers are renamed from the conventional NetFlow labels to short names, so that the data understanding can be improved. As a result of the pre-processing and processing step, the data frame is compressed to 2319 rows, where each row represents a bidirectional flow. There is no anomalous traffic identified, although the flows are flagged as “invalid_flows” because the ToS label is missing from the raw data.

[29] ftp://ftp.bro-ids.org/enterprise-traces/hdr-traces05/

[30] https://www.icir.org/enterprise-tracing/Overview.html

The traffic volume of the top protocols within the network can be visualized in Figure 93. The chart indicates that all the traffic happens over TCP and UDP, almost in equal percentage.

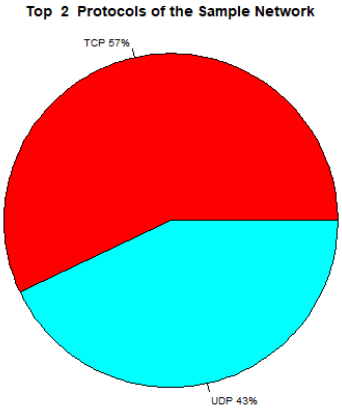

Fig. 93. Top Protocols by Traffic Volume of Network II

The most requested service is SNMP (161), followed by HTTP (80), SMB (445), NBT (139) and Kerberos (88) (Figure 94). SMB[31] (Server Message Block) is used to support shared access to files. SNMP[32] (Simple Network Management Protocol) is used to cluster and set up the information within managed machines. NBT[33] (Netbios) allows applications in a LAN to connect with the hardware and transfer data. Kerberos[34] uses an encryption algorithm to reduce risks in the process of transferring keys. The fact that HTTP belongs to the top protocol validates the proposition 3.

[31] https://docs.microsoft.com/en-us/windows/desktop/fileio/microsoft-smb-protocol-and-cifs-protocol-overview

[32] https://en.wikipedia.org/wiki/Simple_Network_Management_Protocol

[33] https://en.wikipedia.org/wiki/NetBIOS_over_TCP/IP

[34] https://en.wikipedia.org/wiki/Kerberos_(protocol)

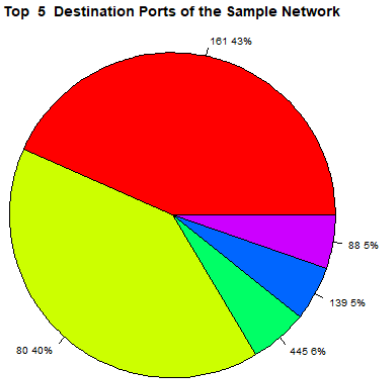

Fig. 94. Top Services of the Network II

The result of profiling the clients is visible in Figure 95.

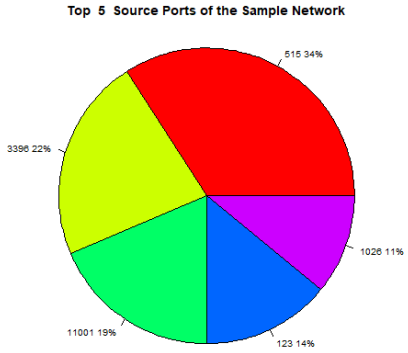

Fig. 95. Top Clients of the Network II

The web servers responsible for more than 1% of the transfer bytes are not all within the same prefix, meaning that they are split over several subnetworks (Figure 96).

DestIP	DestPort
60.244.179.242	80
59.69.194.101	80
220.195.34.140	80
208.133.77.223	80
131.243.160.249	80
131.243.160.249	443
131.243.10.142	80
128.3.2.67	80

Fig. 96. Top Web Servers of the Network II

The web clients sending more than 1% of the transfer bytes use more than one port assigned to the web traffic and are all within the same prefix (131.243.10.x), meaning that they all belong to the same subnetwork (Figure 97). It is also visible that one of the web servers in Figure 96 (namely 131.243.10.142) is assigned to the same subnetwork of the web clients.

SourceIP	DestPort
131.243.10.91	80
131.243.10.91	443
131.243.10.86	80
131.243.10.73	80
131.243.10.73	443
131.243.10.55	80
131.243.10.55	443
131.243.10.26	80
131.243.10.26	443
131.243.10.102	80
131.243.10.10	80

Fig. 97. Top Web Clients of the Network II

Figure 98 shows that there are five DNS servers within the network and that they are all split over several subnetworks.

DestIP	DestPort
59.45.118.136	53
208.14.151.45	53
196.65.236.78	53
128.3.99.102	53
128.3.97.204	53

Fig. 98. Top DNS Servers of the Network II

Most of the DNS clients in Figure 91 are also web clients (Figure 89) (proposition 17) and remote clients (Figure 99) and they belong to the same subnetwork assigned to the web clients (131.243.10.x).

SourceIP	DestPort
131.243.10.92	53
131.243.10.91	53
131.243.10.86	53
131.243.10.55	53
131.243.10.41	53
131.243.10.209	53
131.243.10.203	53
131.243.10.152	53
131.243.10.148	53
131.243.10.102	53

Fig. 99. Top DNS Clients of the Network II

The mail servers (Figure 100) are split over several subnetworks and use various mail ports: 25, 110, 993 and 995.

DestIP	DestPort
208.29.20.162	25
208.29.20.162	110
204.215.97.78	993
131.243.10.203	25
128.3.164.248	25
128.3.164.194	993
128.3.164.15	993
128.3.161.107	993
128.3.161.107	995

Fig. 100. Top Mail Servers of the Network II

There is a large amount of mail clients (Figure 101) and several of them are web, remote and DNS clients, as well.

SourceIP	DestPort
131.243.10.91	993
131.243.10.91	25
131.243.10.87	993
131.243.10.87	995
131.243.10.86	25
131.243.10.86	110
131.243.10.73	993
131.243.10.73	25
131.243.10.55	993
131.243.10.55	25
131.243.10.37	993
131.243.10.26	993
131.243.10.243	993
131.243.10.217	993
131.243.10.172	993
131.243.10.171	993
131.243.10.159	993
131.243.10.148	993
131.243.10.132	993
131.243.10.123	993
131.243.10.106	993
131.243.10.106	25
131.243.10.10	993
128.3.164.249	25
128.3.164.248	25

Fig. 101. Top Mail Clients of the Network II

The remote servers (Figure 102) run over destination port 22 and are split over several subnetworks.

DestIP	DestPort
128.3.193.70	22
128.55.120.191	22
128.55.52.190	22
128.55.56.195	22
128.55.97.217	22
170.228.197.133	22
170.228.197.92	22

Fig. 102. Top Remote Servers of the Network II

There are four remote clients (Figure 103) running over several high ports, which are all within the same subnetwork (131.243.10.x).

SourceIP	SourcePort
131.243.10.143	58626
131.243.10.253	1525
131.243.10.253	1528
131.243.10.253	1526
131.243.10.253	1527
131.243.10.50	32833
131.243.10.50	32771
131.243.10.50	32834
131.243.10.50	32831
131.243.10.50	32832
131.243.10.91	32842
131.243.10.91	32839
131.243.10.91	32840
131.243.10.91	32841

Fig. 103. Top Remote Clients of the Network II

This network sample includes all services in scope except VPN and in comparison with Network I, it also has remote and mail services. The P2P algorithm found 47 P2P flows out of 2319, which means that roughly 2% of the total traffic is P2P traffic. 718 out of 2319 flows are marked as “Attack/PortScan” according to the methodology outlined in Section 3.9.2 and roughly half of the flows are flagged as “one-to-several:browsing”, while the rest of the flows is flagged as “one-to-one:download”. Approximately half of the flows are marked as “one-to-many:netscan”. This indicates that the network traffic requires a closer look-up to be able to conclude if a network attack is existent or not.

The network is analysed along the five heuristics applied to the vertices (Figures 104, 105 and 106).

ID	Label	ClientServer	RemoteClientServer	Mail	DNS	TotalTransfer	SuspectFlow	DoS	Worms_Trojans	possibleScan	VPN	HostType
1	131.243.10.10	webClient	remoteHost	mailClient	DNSHost	78291.6213	invalid_Tos	NA	NA		NA	P2PNONP2P
2	128.3.161.197	webHost	remoteHost	mailHost	DNSHost	79363.9851	invalid_Tos	NA	NA		NA	P2PNONP2P
3	128.3.161.62	webHost	remoteHost	mailHost	DNSHost	4407.9375	invalid_Tos	NA	NA		NA	P2P
4	128.3.164.191	webHost	remoteHost	mailHost	DNSHost	2656.7092	invalid_Tos	NA	NA		NA	P2P
5	131.243.10.102	webClient	remoteHost	mailHost	DNSClient	9210.0183	invalid_Tos	NA	NA		NA	NONP2P
6	131.243.10.106	webClient	remoteHost	mailClient	DNSHost	19498.6920	invalid_Tos	NA	NA		NA	P2PNONP2P
7	128.3.161.95	webHost	remoteHost	mailHost	DNSHost	1319.8598	invalid_Tos	NA	NA		NA	P2P
8	131.243.10.108	webClient	remoteHost	mailHost	DNSHost	12981.4912	invalid_Tos	NA	NA		NA	NONP2P
9	131.243.10.116	webHost	remoteHost	mailHost	DNSHost	19985.8805	invalid_Tos	NA	NA		NA	P2PNONP2P
10	131.243.10.122	webHost	remoteHost	mailHost	DNSHost	1292.0000	invalid_Tos	NA	NA		NA	NONP2P
11	131.243.10.123	webHost	remoteHost	mailClient	DNSHost	1578.4873	invalid_Tos	NA	NA		NA	NONP2P
12	131.243.10.131	webHost	remoteHost	mailHost	DNSHost	180.0000	invalid_Tos	NA	NA		NA	NONP2P
13	77.47.14.18	webHost	remoteHost	mailHost	DNSHost	38345.7129	invalid_Tos	NA	NA	possibleScan	NA	NONP2P
14	131.243.10.132	webClient	remoteHost	mailClient	DNSHost	48024.9617	invalid_Tos	NA	NA		NA	NONP2P
15	128.3.212.145	webHost	remoteHost	mailHost	DNSHost	10032.0000	invalid_Tos	NA	NA	possibleScan	NA	NONP2P
16	128.3.48.249	webHost	remoteHost	mailHost	DNSHost	4224.0000	invalid_Tos	NA	NA		NA	NONP2P
17	128.3.96.230	webHost	remoteHost	mailHost	DNSHost	15664.0000	invalid_Tos	NA	NA	possibleScan	NA	NONP2P
18	131.243.10.142	webServer	remoteHost	mailHost	DNSHost	229969.5286	invalid_Tos	NA	NA		NA	P2PNONP2P
19	128.3.188.91	webHost	remoteHost	mailHost	DNSHost	933.5674	invalid_Tos	NA	NA		NA	NONP2P

Fig. 104. Vertex Attributes of the IP Network II

ID	Label	ClientServer	RemoteClientServer	Mail	DNS	TotalTransfer	SuspectFlow	DoS	Worms_Trojans	possibleScan	VPN	HostType
53	146.17.26.88 0	webClient	remoteHost	mailHost	DNSHost	164.0000	NA	NA	NA	NA	VPN	NONP2P
54	146.17.26.89 0	webClient	remoteHost	mailHost	DNSHost	300.9252	NA	NA	NA	NA	VPN	NONP2P
55	146.17.26.92 0	webClient	remoteHost	mailHost	DNSClient	2481.6350	NA	NA	NA	NA	VPN	NONP2P
56	146.16.251.99 5061	webHost	remoteHost	mailHost	DNSHost	7138.3323	NA	NA	NA	NA	NA	P2P
57	146.16.250.103 23018	webHost	remoteHost	mailHost	DNSHost	408.0000	NA	NA	NA	NA	NA	P2P
58	146.16.250.103 23019	webHost	remoteHost	mailHost	DNSHost	180.0000	NA	NA	NA	NA	NA	P2P
59	146.17.25.33 43397	webHost	remoteHost	mailHost	DNSHost	283.0000	NA	NA	NA	NA	NA	NONP2P
60	146.16.250.103 31354	webHost	remoteHost	mailHost	DNSHost	408.0000	NA	NA	NA	NA	NA	P2P
61	146.16.250.103 31355	webHost	remoteHost	mailHost	DNSHost	180.0000	NA	NA	NA	NA	NA	P2P
62	146.17.25.36 49760	webHost	remoteHost	mailHost	DNSHost	283.0000	NA	NA	NA	NA	NA	NONP2P
63	146.16.120.111 5012	webHost	remoteHost	mailHost	DNSHost	428.0000	NA	NA	NA	NA	NA	P2P
64	146.16.120.111 5013	webHost	remoteHost	mailHost	DNSHost	196.0000	NA	NA	NA	NA	NA	P2P
65	146.17.25.43 47388	webHost	remoteHost	mailHost	DNSHost	283.0000	NA	NA	NA	NA	NA	NONP2P
66	146.17.25.48 32948	webHost	remoteHost	mailHost	DNSHost	283.0000	NA	NA	NA	NA	NA	NONP2P
67	146.17.223.220 5014	webHost	remoteHost	mailHost	DNSHost	428.0000	NA	NA	NA	NA	NA	P2P
68	146.17.223.220 5015	webHost	remoteHost	mailHost	DNSHost	196.0000	NA	NA	NA	NA	NA	P2P
69	146.17.25.62 60409	webHost	remoteHost	mailHost	DNSHost	283.0000	NA	NA	NA	NA	NA	NONP2P
70	146.17.223.154 5010	webHost	remoteHost	mailHost	DNSHost	428.0000	NA	NA	NA	NA	NA	P2P
71	146.17.223.154 5011	webHost	remoteHost	mailHost	DNSHost	196.0000	NA	NA	NA	NA	NA	P2P
72	146.16.250.119 23454	webHost	remoteHost	mailHost	DNSHost	408.0000	NA	NA	NA	NA	NA	P2P
73	146.16.250.119 23455	webHost	remoteHost	mailHost	DNSHost	180.0000	NA	NA	NA	NA	NA	P2P
74	146.17.25.82 45529	webHost	remoteHost	mailHost	DNSHost	283.0000	NA	NA	NA	NA	NA	NONP2P

Fig. 105. Vertex Attributes of the Pair Network II

ID	Label	TotalTransfer	HostType
1	1072	4486.3889	P2PNONP2P
2	1087	1651.4437	NONP2P
3	3063	383.7270	NONP2P
4	3131	386.1020	NONP2P
5	3193	387.7895	NONP2P
6	3255	387.7895	NONP2P
7	3384	387.7895	NONP2P
8	2243	946.5556	NONP2P
9	2290	946.1659	NONP2P
10	2338	970.2101	NONP2P
11	2378	975.6948	NONP2P
12	2428	2529.6747	NONP2P
13	1063	1164.9207	NONP2P
14	1064	813.7852	P2PNONP2P
15	1088	1153.3875	P2PNONP2P
16	1033	5680.2077	P2PNONP2P
17	2227	719.9167	P2PNONP2P

Fig. 106. Vertex Attributes of the Port Network II

Figure 107 provides a visualization of the network from the point of view of Client/Server interaction, where the clients have the color blue, the servers are green and the rest of the hosts are red. It is visible that this graph is split into several components, thus it is disconnected, which corresponds to the proposition 1. As noticeable in the Figure 96 and in Figure 97, the web servers and the web clients are mainly included in a single subnetwork, which is also visible in the cluster on the left side of the graph in Figure 107, which contains all the web servers and all the web clients (proposition 4).

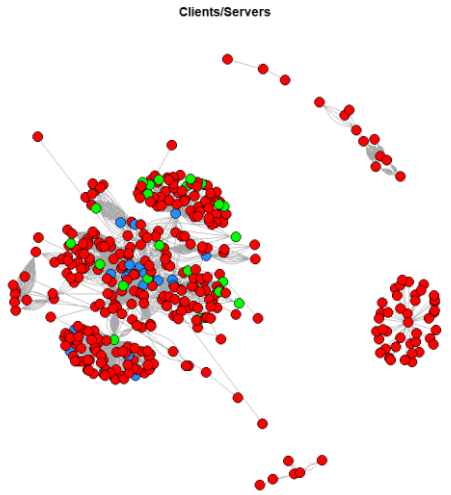

Fig. 107. Network II Visualization Clients/Servers

Figure 108 provides a visualization of the network from the point of view of DNS interaction, where the DNS clients have the color blue, the DNS servers are green and the rest of the hosts are red. Even if the DNS servers have distinct network prefixes (Figure 98), they are all assigned to the same cluster (Figure 108) – proposition 4. The DNS clients are also included in the same cluster on the left side of the graph in Figure 108.

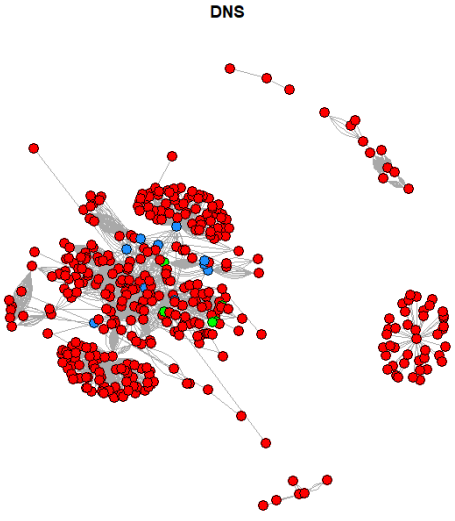

Fig. 108. Network II Visualization DNS

Figure 109 provides a visualization of the network from the perspective of mail interaction, where the mail clients have the color blue, the mail servers are green and the rest of the hosts are red. Even if the mail servers have distinct network prefixes (Figure 100), they all belong to the same cluster (Figure 109), which confirms that the servers are all placed within a cluster independently from their prefix assignment – proposition 4. The mail clients are also included in the same cluster situated on the left side of the graph in Figure 109.

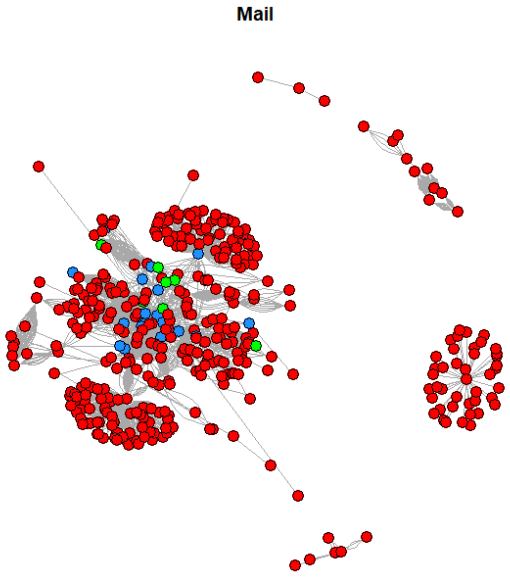

Fig. 109. Network II Visualization Mail

Figure 110 provides a visualization of the network from the perspective of P2P interaction, where the NONP2P hosts have the color blue, the hosts being both P2P and NONP2P are green and P2P hosts are red. The same as in case of Network I, this graph yields a cluster containing exclusively P2P connections – proposition 2 (the cluster on the right side of Figure 110).

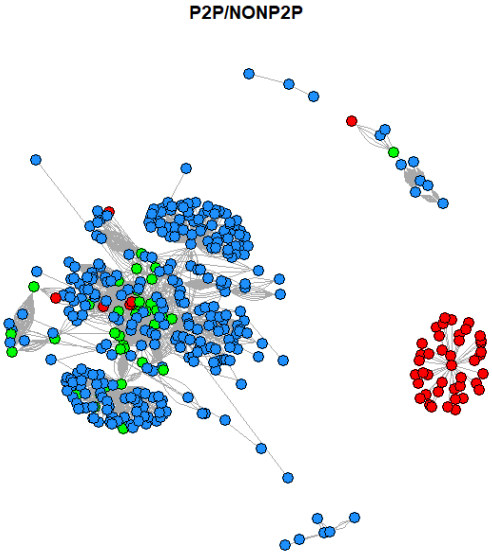

Fig. 110. Network II Visualization P2P Traffic

Figure 111 provides a visualization of the network from the perspective of remote interaction, where the remote servers have the color green, the remote clients are blue and the remote hosts are red.

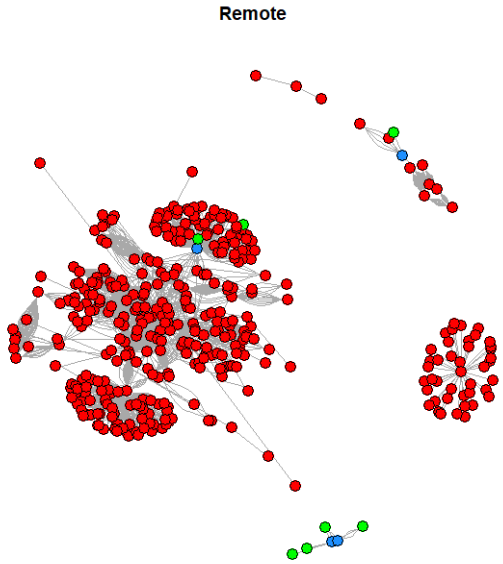

Fig. 111. Network II Visualization remote Traffic

The largest egocentric subgraphs can be visualized below in Figure 112, Figure 113, Figure 114, Figure 115 and Figure 116, where the color encoding is the same as for Network I. A web server is again among the top five largest subgraphs of Network II (proposition 5), while the rest of the neighborhoods are built around clients and hosts.

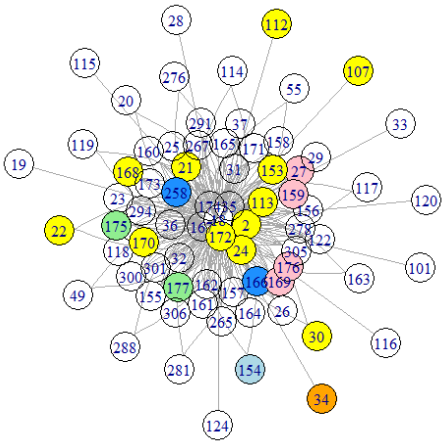

Fig. 112. First Order Neighborhood Network II - Rank 1

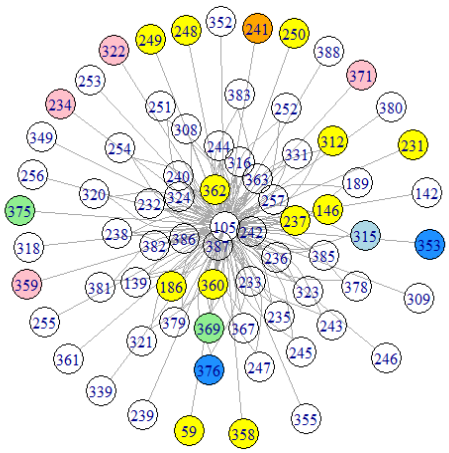

Fig. 113. First Order Neighborhood Network II - Rank 2

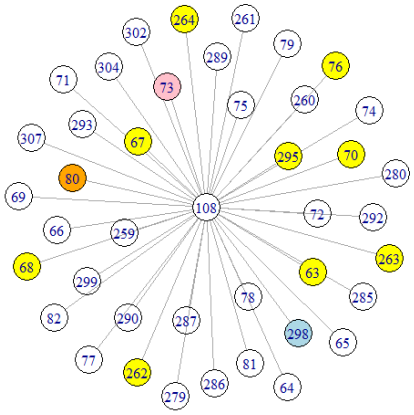

Fig. 114. First Order Neighborhood Network II - Rank 3

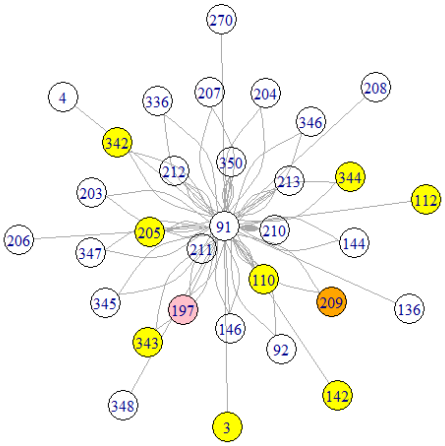

Fig. 115. First Order Neighborhood Network II - Rank 4

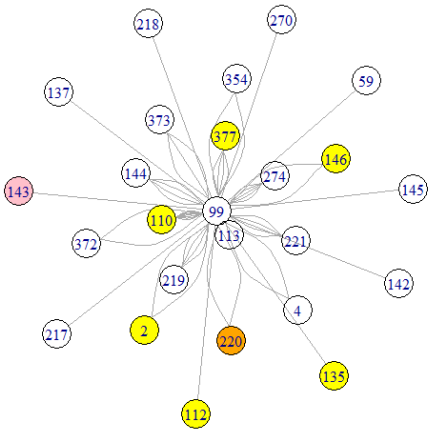

Fig. 116. First Order Neighborhood Network II - Rank 5

The largest egocentric subgraph of the Port Network contains the port 80 (ID 1.379) and its 651 direct neighbors (Figure 117). Port 80 defines HTTP traffic, which explains its high amount of neighbors and validates also the findings from the analysis of the top egocentric subgraphs of the IP Network, where a host labeled as web server is in top 5 biggest first-order neighborhoods. This pattern could be identified also in the first data set (proposition 3).

NeighborsCount	ID
651	1379
325	1375
93	1383
92	1377
76	1388

Fig. 117. First Order Neighborhood Port Network II

The largest egocentric subgraph of the Pair Network contains the pair (128.3.161.74, 80) (ID 1698) and its 173 direct neighbors (Figure 118). This pair defines again a web server.

NeighborsCount	ID
173	1698
87	1927
85	1722
78	1761
78	1767
78	1769

Fig. 118. First Order Neighborhood Pair Network II

As expected, the largest egocentric subgraph of the PairToIP Network contains again the IP 128.3.161.74 and its 175 direct neighbors (Figure 119).

NeighborsCount	ID
175	2358
126	2238
101	2330
87	2434
85	2372

Fig. 119. First Order Neighborhood PairToIP Network II

The largest egocentric subgraph of the PairToPort Network contains the port 80 and its 680 direct neighbors (Figure 120).

NeighborsCount	ID
680	3599
332	3595
173	1698
97	3603
93	3597

Fig. 120. First Order Neighborhood PairToPort Network II

The degree distribution of all network types can be visualized in the Figures 121, 122, 123, 124 and 125. It is visible that the results are analogous to the ones corresponding to Network I: most of the nodes have a degree between 0 and 50 (proposition 6), while there are only a few nodes having high degrees (proposition 18).

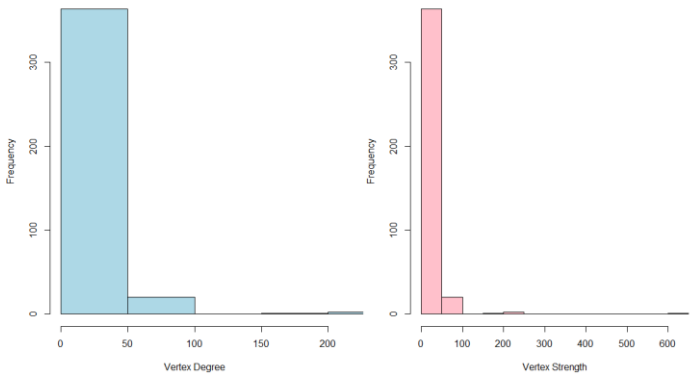

Fig. 121. Degree Distribution (left) and strength (right) of IP Network II

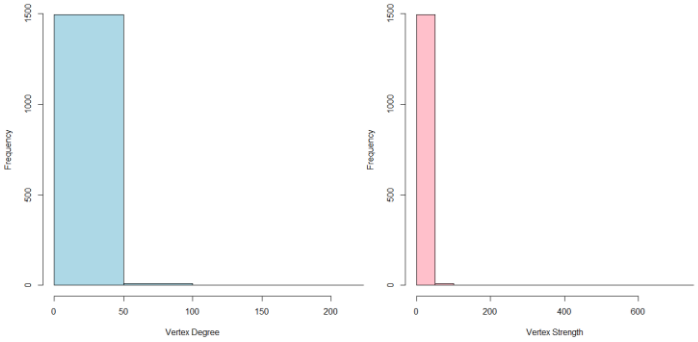

Fig. 122. Degree Distribution (left) and strength (right) of Port Network II

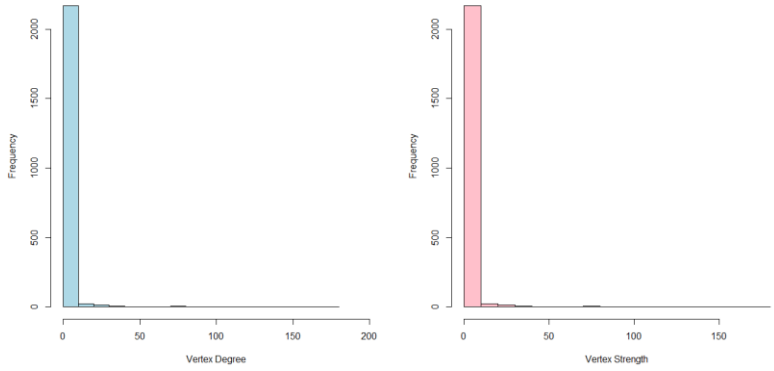

Fig. 123. Degree Distribution (left) and strength (right) of Pair Network II

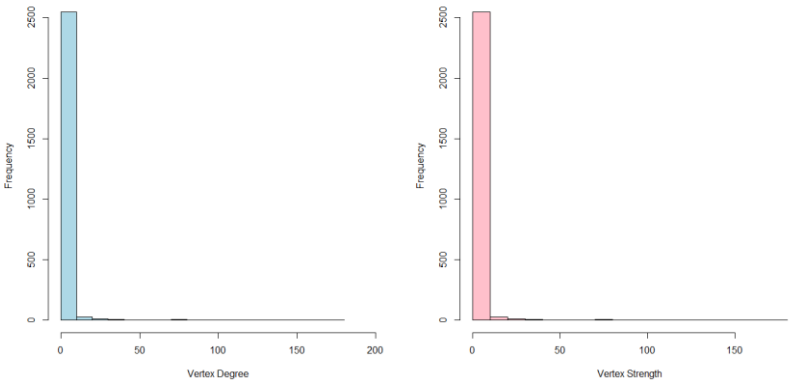

Fig. 124. Degree Distribution (left) and strength (right) of PairToIP Network II

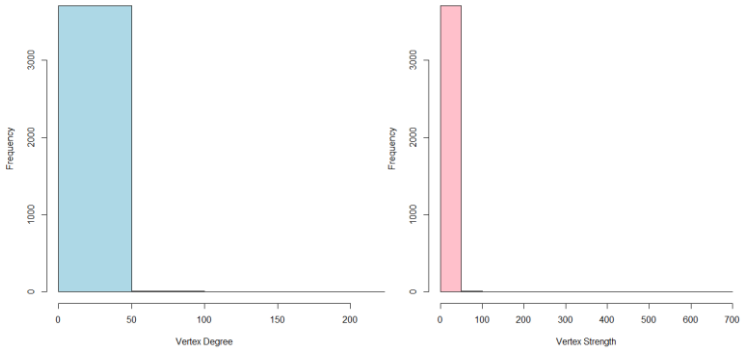

Fig. 125. Degree Distribution (left) and strength (right) of PairToPort Network II

The average neighbor degree (Figures 126, 127, 128, 129 and 130) shows the same pattern as the one identified in the first data set: a tendency for vertices having low degrees to connect to vertices of rather higher degrees – proposition 7. This confirms also the behavior "one-to-several:browsing" – proposition 8.

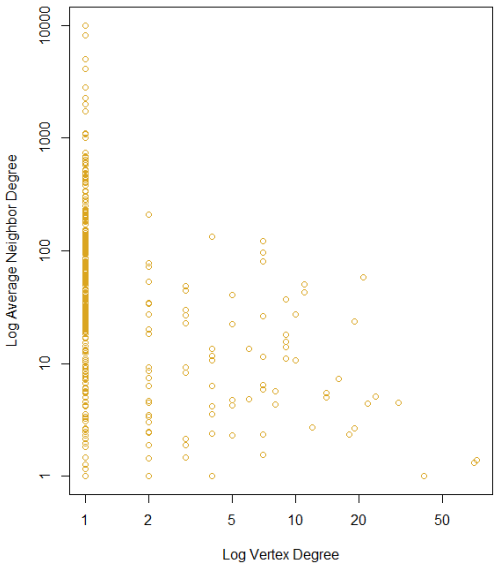

Fig. 126. Average Neighbor Degree versus Vertex Degree of the IP Network II

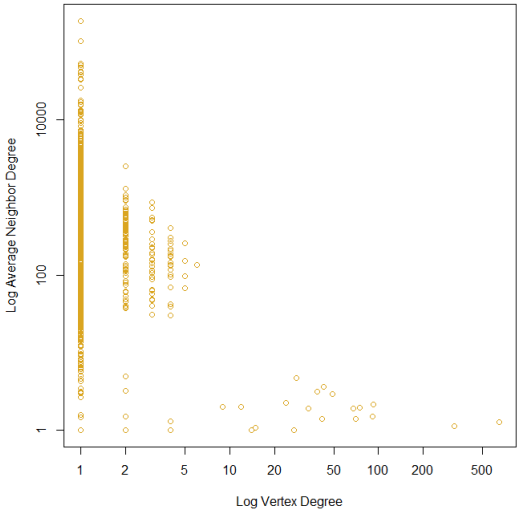

Fig. 127. Average Neighbor Degree versus Vertex Degree of the Port Network II

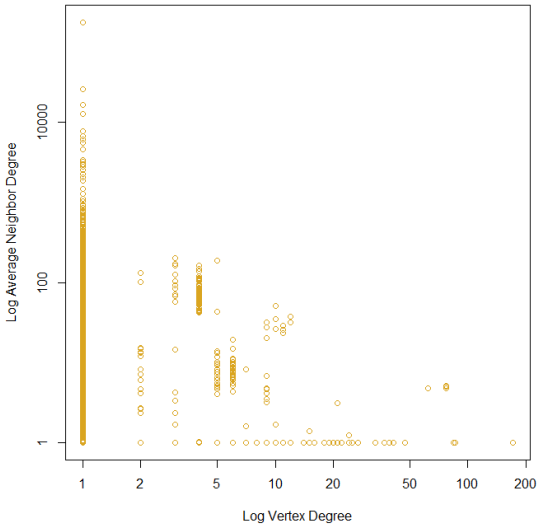

Fig. 128. Average Neighbor Degree versus Vertex Degree of the Pair Network II

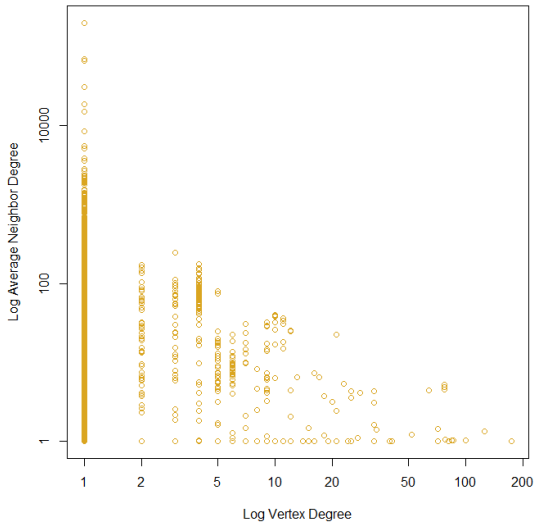

Fig. 129. Average Neighbor Degree versus Vertex Degree of the PairToIP Network II

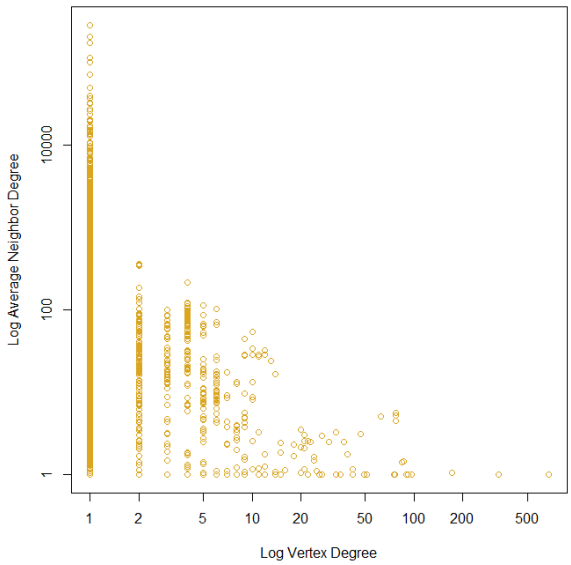

Fig. 130. Average Neighbor Degree versus Vertex Degree PairToPort Network II

All connected components within each graph belonging to one of the 5 graph types are enumerated in Figures 131, 132, 133 and 134. This network is also disconnected, since the IP Network contains six components, and also that there is a giant component, which contains 323 of the total vertices (388) of the graph (Figure 131). There are 37 articulation points within the giant component (proposition 9), which corresponds to roughly 10% of the total vertices. Recall that a secure network aims to decrease this percentage as much as possible, so it is indicated to monitor the hosts enumerated in Figure 132.

```
> comps <- decompose.graph(net_IP)
> table(sapply(comps, vcount))

  2   3   5   6  42 323
  2   1   2   1   1   1
```

Fig. 131. Connected Components of IP Network II

```
> net_IP.cut.vertices <- articulation.points(net_IP.gc)
> net_IP.cut.vertices
+ 37/323 vertices, from 1fa0c9b:
 [1]  39  14  35  82  55   5  67 117 110  10 111  13  63  73  66   8 144  60  50  76 317  74   9 109  70  62  51   3  58  64  48  97  46  43  37   6   1
> length(net_IP.cut.vertices)
[1] 37
```

Fig. 132. Cut Vertices of IP Network II

The Port Network yields also a giant connected component as in the case of Network I, as illustrated in Figure 133. This component contains 1449 vertices, meaning that the endpoints of communication also tend to be interconnected.

```
> comps <- decompose.graph(net_Port)
> table(sapply(comps, vcount))

   1    2    3    5   15   28 1449
   4    1    1    1    1    1    1
```

Fig. 133. Connected Components of Port Network II

There are three giant components of the Pair Network II (Figure 134), each of them containing 141, 171 and respectively, 173 vertices. There are also various other components and can be noticed that the highest number of all components (75) has again the rank 2 from the perspective of (IP, port) heuristics. This means there are several interconnected (IP, port) pairs, which are disconnected from the rest of the network.

```
> comps <- decompose.graph(net_IP_Pair)
> table(sapply(comps, vcount))

  2   3   4   5   6   7   8   9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  25  26  28  34  36  38  40  42  48 141 171 173
 75  57  26  15  17   6   9   9   7   2   5   2   1   1   2   1   1   2   1   3   3   2   1   3   2   2   1   1   1   1   1   1   1   1
```

Fig. 134. Connected Components of Pair Network II

The PairToIP Network (Figure 135) yields a giant component of 428 vertices, meaning that like in the case of Network I, the majority of the (IP, port) pairs tend to link to the same IP, respectively the majority of the IPs tend to connect to the same (IP, port) pair (proposition 12).

```
> comps <- decompose.graph(net_PairToIP)
> table(sapply(comps, vcount))

  2   3   4   5   6   7   8   9  10  11  12  13  14  15  16  17  20  22  25  26  34  41  42  46  63  72 156 171 175 305 428
 61  53  22  13  15   6   9   3   5   1   2   3   1   1   1   1   1   2   2   1   1   1   3   1   1   1   1   1   1   1   1
```

Fig. 135. Connected Components of PairToIP Network II

The PairToPort Network (Figure 136) has one giant component containing 1168 vertices, meaning that most of the (IP, port) pairs tend to link to the same port, respectively the ports tend to connect to the same (IP, port) pair (proposition 13).

```
> comps <- decompose.graph(net_PairPort)
> table(sapply(comps, vcount))

   2    3    4    5    6    7    8    9   10   12   13   15   16   23   27   28   34   36   43   50   52   76   77   90   93   97  102  332  680
  58   45   17    8    8    6    4    2    4    1    1    1    2    1    1    2    1    2    2    1    1    1    1    1    1    1    1    1    1
1168
   1
```

Fig. 136. Connected Components of PairToPort Network II

Figure 137, Figure 138, Figure 139, Figure 140 and Figure 141 highlight the community size for each of the 5 network types. It is visible that none of the networks have a community that dominates in size all the others and the IP Network has the lowest number of communities (18 communities), similar to the Network I.

Only a few nodes of the Pair Network (Figure 139), PairToIP Network (Figure 140) and PairToPort Network (Figure 141) are interconnected in terms of membership, while the Port Network (Figure 138) contains more interlinked vertices than in case of the first data set.

```
> length(kc)
[1] 18
> sizes(kc)
Community sizes
 1  2  3  4  5  6  7  8  9 10 11 12 13 14 15 16 17 18
36 39 14 68 37 69 10 38  6 42  6  5  4  3  2  2  5  2
```

Fig. 137. Community Size IP Network II

```
> length(kc)
[1] 20
> sizes(kc)
Community sizes
  1   2   3   4   5   6   7   8   9  10  11  12  13  14  15  16  17  18  19  20
589 289 104  62 145 109  41  15  70   9  16  28  15   5   3   2   1   1   1   1
```

Fig. 138. Community Size Port Network II

```
> length(kc)
[1] 266
> sizes(kc)
Community sizes
  1   2   3   4   5   6   7   8   9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28  29  30  31  32  33  34  35  36  37
101 173  85  86  18  48  42  40  38  34  34  36  28  28  10  26  26  26  25  14  23  23  22  22  22  21  21  21  20  27  19  19   8  17  16  16  15
 38  39  40  41  42  43  44  45  46  47  48  49  50  51  52  53  54  55  56  57  58  59  60  61  62  63  64  65  66  67  68  69  70  71  72  73  74
 13  13  13   6  12  12  12  12  12  11  11  10  10  10  10  10  10   9   9   9   9   9   9   9   9   8   8   8   8   8   8   8   8   7   7   7
 75  76  77  78  79  80  81  82  83  84  85  86  87  88  89  90  91  92  93  94  95  96  97  98  99 100 101 102 103 104 105 106 107 108 109 110 111
  7   7   7   6   6   6   6   6   6   6   6   6   6   6   6   6   6   6   5   5   5   5   5   5   5   5   5   5   5   5   5   5   4   4   4
112 113 114 115 116 117 118 119 120 121 122 123 124 125 126 127 128 129 130 131 132 133 134 135 136 137 138 139 140 141 142 143 144 145 146 147 148
  4   4   4   4   4   4   4   4   4   4   4   4   4   4   4   4   4   4   4   4   4   4   4   3   3   3   3   3   3   3   3   3   3   3   3   3   3
149 150 151 152 153 154 155 156 157 158 159 160 161 162 163 164 165 166 167 168 169 170 171 172 173 174 175 176 177 178 179 180 181 182 183 184 185
  3   3   3   3   3   3   3   3   3   3   3   3   3   3   3   3   3   3   3   3   3   3   3   3   3   3   3   3   3   3   3   3   3   3   3   3   3
186 187 188 189 190 191 192 193 194 195 196 197 198 199 200 201 202 203 204 205 206 207 208 209 210 211 212 213 214 215 216 217 218 219 220 221 222
  3   3   3   3   3   3   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2
223 224 225 226 227 228 229 230 231 232 233 234 235 236 237 238 239 240 241 242 243 244 245 246 247 248 249 250 251 252 253 254 255 256 257 258 259
  2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2
260 261 262 263 264 265 266
  2   2   2   2   2   2   2
```

Fig. 139. Community Size Pair Network II

```
> length(kc)
[1] 226
> sizes(kc)
Community sizes
  1   2   3   4   5   6   7   8   9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28  29  30  31  32  33  34  35  36
191  42  63 116 119  74 101 175  85  82  46 122  86  72  32  42  42  41  34  26  25  25  14  22  22  20  27   8   8  17  16  15  13   6  13  13
 37  38  39  40  41  42  43  44  45  46  47  48  49  50  51  52  53  54  55  56  57  58  59  60  61  62  63  64  65  66  67  68  69  70  71  72
 13  12  12   6  11  10  10  10  10  10   9   9   9   8   8   8   8   8   8   8   7   7   7   7   7   6   6   6   6   6   6   6   6   6   6
 73  74  75  76  77  78  79  80  81  82  83  84  85  86  87  88  89  90  91  92  93  94  95  96  97  98  99 100 101 102 103 104 105 106 107 108
  6   6   6   6   6   5   5   5   5   5   5   5   5   5   5   5   5   5   4   4   4   4   4   4   4   4   4   4   4   4   4   4   4   4   4   4
109 110 111 112 113 114 115 116 117 118 119 120 121 122 123 124 125 126 127 128 129 130 131 132 133 134 135 136 137 138 139 140 141 142 143 144
  4   4   4   3   3   3   3   3   3   3   3   3   3   3   3   3   3   3   3   3   3   3   3   3   3   3   3   3   3   3   3   3   3   3   3   3
145 146 147 148 149 150 151 152 153 154 155 156 157 158 159 160 161 162 163 164 165 166 167 168 169 170 171 172 173 174 175 176 177 178 179 180
  3   3   3   3   3   3   3   3   3   3   3   3   3   3   3   3   3   3   3   3   3   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2
181 182 183 184 185 186 187 188 189 190 191 192 193 194 195 196 197 198 199 200 201 202 203 204 205 206 207 208 209 210 211 212 213 214 215 216
  2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2
217 218 219 220 221 222 223 224 225 226
  2   2   2   2   2   2   2   2   2   2
```

Fig. 140. Community Size PairToIP Network II

```
> length(kc)
[1] 192
> sizes(kc)
Community sizes
  1   2   3   4   5   6   7   8   9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28  29  30  31  32  33  34  35  36
113 184 110 296  75 680 190  32 332  38  97  93  90  77  76  72  52  50  43  43  42  21  36  36  34   9  30  28  28  27  14  23  26   8  16  16
 37  38  39  40  41  42  43  44  45  46  47  48  49  50  51  52  53  54  55  56  57  58  59  60  61  62  63  64  65  66  67  68  69  70  71  72
 15   8  13  13  12   6  10  10  10  10   5   9   8   8   8   7   7   7   7   7   7   6   6   6   6   6   6   6   5   5   5   5   5   5   5   5
 73  74  75  76  77  78  79  80  81  82  83  84  85  86  87  88  89  90  91  92  93  94  95  96  97  98  99 100 101 102 103 104 105 106 107 108
  4   4   4   4   4   4   4   4   4   4   4   4   4   4   4   4   4   3   3   3   3   3   3   3   3   3   3   3   3   3   3   3   3   3   3   3
109 110 111 112 113 114 115 116 117 118 119 120 121 122 123 124 125 126 127 128 129 130 131 132 133 134 135 136 137 138 139 140 141 142 143 144
  3   3   3   3   3   3   3   3   3   3   3   3   3   3   3   3   3   3   3   3   3   3   3   3   3   2   2   2   2   2   2   2   2   2   2   2
145 146 147 148 149 150 151 152 153 154 155 156 157 158 159 160 161 162 163 164 165 166 167 168 169 170 171 172 173 174 175 176 177 178 179 180
  2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2
181 182 183 184 185 186 187 188 189 190 191 192
  2   2   2   2   2   2   2   2   2   2   2   2
```

Fig. 141. Community Size PairToPort Network II

The visual representations resulted by plotting the IP Network (Figure 142), the Pair Network (Figure 143) and the Port Network (Figure 144) provide a good overview to comprehend how the communities are spread within the networks and how they connect to each other.

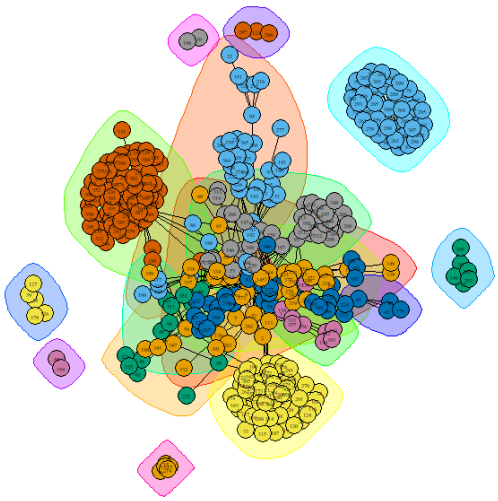

Fig. 142. Community Visualization IP Network II

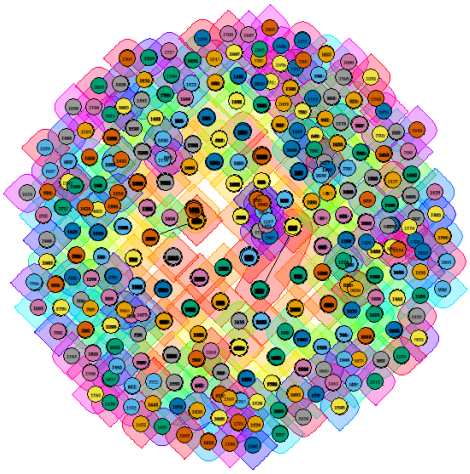

Fig. 143. Community Visualization Pair Network II

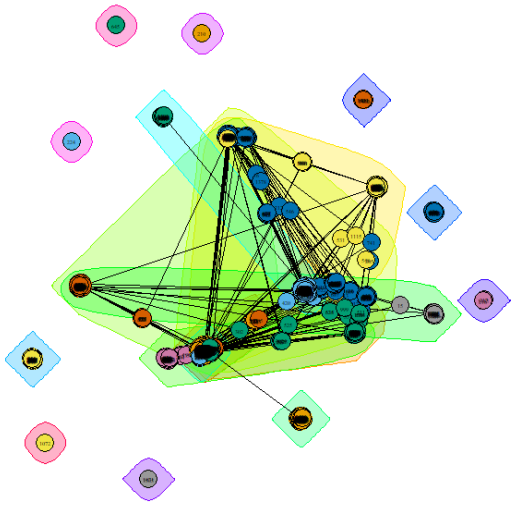

Fig. 144. Community Visualization Port Network II

The evaluation of functional classes exhibits the same pattern as in case of Network I. The nodes are spread over all functional classes for the IP Network, the Port Network and the Pair Network, which makes their assigned communities less explainable against the validation sets – proposition 10.

DNSClient	DNSHost	DNSServer	webClient	webHost	webServer	remoteClient	remoteHost	remoteServer	mailClient	mailHost	mailServer	invalid_Tos		possibleScan	NONP2P	P2P	P2PNONP2P
2	34	0	4	29	3	0	36	0	6	27	3	36	35	1	28	2	6
6	28	5	3	33	3	0	39	0	2	36	1	39	39	0	36	1	2
0	14	0	0	13	1	0	14	0	1	13	0	14	14	0	8	0	6
0	68	0	5	62	1	0	68	0	0	68	0	68	68	0	62	0	6
1	36	0	2	34	1	0	37	0	3	33	1	37	37	0	27	0	10
1	68	0	1	59	9	1	66	2	1	67	1	69	69	0	69	0	0
0	10	0	1	8	1	0	10	0	1	9	0	10	10	0	9	0	1
0	38	0	2	33	3	0	38	0	3	35	0	38	38	0	32	2	4
0	6	0	1	4	1	0	6	0	0	6	0	6	6	0	6	0	0
0	42	0	0	42	0	0	42	0	0	42	0	42	42	0	0	42	0
0	6	0	0	6	0	2	0	4	0	6	0	6	6	0	6	0	0
0	5	0	0	5	0	1	3	1	0	5	0	5	5	0	3	1	1
0	4	0	0	4	0	0	4	0	0	4	0	4	4	0	4	0	0
0	3	0	0	3	0	0	3	0	0	3	0	3	3	0	3	0	0
0	2	0	0	2	0	0	2	0	0	2	0	2	2	0	2	0	0
0	2	0	0	2	0	0	2	0	0	2	0	2	2	0	2	0	0
0	5	0	0	5	0	0	5	0	0	5	0	5	1	4	5	0	0
0	2	0	0	2	0	0	2	0	1	0	1	2	2	0	1	0	1

Fig. 145. Functional Classes IP Network II

```
> table(c.m, func.class,  useNA=c("no"))
    func.class
c.m  NONP2P P2P P2PNONP2P
  1      580   0         9
  2      286   0         3
  3       70   6        28
  4       56   2         4
  5      108  22        15
  6      107   0         2
  7       41   0         0
  8        4   8         3
  9       34  31         5
  10       8   0         1
  11       0  15         1
  12      28   0         0
  13      15   0         0
  14       0   5         0
  15       3   0         0
  16       0   2         0
  17       1   0         0
  18       1   0         0
  19       0   1         0
  20       1   0         0
```

Fig. 146. Functional Classes Port Network II

DNSClient	DNSHost	DNSServer	webClient	webHost	webServer	remoteClient	remoteHost	remoteServer	mailClient	mailHost	mailServer	invalid_Tos		possibleScan	NONP2P	P2P
7	19	0	22	3	1	3	23	0	18	8	0	26	26	0	26	0
2	24	0	9	17	0	0	26	0	9	17	0	26	26	0	26	0
0	25	0	9	16	0	0	25	0	9	16	0	25	25	0	25	0
0	14	0	0	14	0	0	14	0	0	14	0	14	14	0	14	0
0	23	0	2	20	1	0	23	0	0	23	0	23	23	0	23	0
14	9	0	22	1	0	0	23	0	22	1	0	23	23	0	23	0
21	1	0	21	0	1	0	22	0	21	1	0	22	22	0	22	0
0	22	0	21	1	0	0	22	0	21	1	0	22	22	0	22	0
0	22	0	0	21	1	0	22	0	0	22	0	22	22	0	22	0
0	21	0	9	12	0	0	21	0	9	12	0	21	21	0	21	0
0	21	0	16	5	0	0	21	0	16	5	0	21	21	0	21	0
0	21	0	3	18	0	0	21	0	2	19	0	21	21	0	21	0
5	15	0	15	5	0	0	20	0	19	0	1	20	20	0	20	0
0	27	0	15	12	0	0	27	0	0	27	0	27	27	0	27	0
6	13	0	13	6	0	0	19	0	13	6	0	19	19	0	19	0
6	13	0	14	5	0	0	19	0	14	5	0	19	19	0	19	0
6	0	2	6	2	0	0	8	0	6	2	0	8	8	0	8	0
0	17	0	16	0	1	0	17	0	16	1	0	17	17	0	17	0

Fig. 147. Functional Classes Pair Network II

The results of the correlation between the continuous variables (Figure 148, Figure 149, Figure 150, Figure 151 and Figure 152) exhibit the same patterns as in case of Network I. There is a high correlation between betweenness and degree (proposition 11), between degree and total transfer in bytes (proposition 14) and between degree and neighbors (proposition 15) over all network types. The low values of the correlation between closeness and all the other variables is the result of applying again the closeness measure on disconnected networks, which does not provide an accurate estimation. The correlation between the duration in seconds and degree is again below 50% (proposition 16), which indicates that a higher number of links between two adjacent vertices does not stand for a higher duration of the transfer, but it is also possible that vertices connected by a few links send over higher amounts of time.

```
> cor(nodes_IP$Betweenness,nodes_IP$Degree)
[1] 0.6713705
> cor(nodes_IP$Betweenness,nodes_IP$Community)
[1] -0.09834549
> cor(nodes_IP$Betweenness,nodes_IP$TotalTransfer)
[1] 0.8315515
> cor(nodes_IP$Degree,nodes_IP$Community)
[1] -0.003231809
> cor(nodes_IP$Degree,nodes_IP$TotalTransfer)
[1] 0.8879085
> cor(nodes_IP$Neighbors,nodes_IP$TotalTransfer)
[1] 0.8299461
> cor(nodes_IP$Neighbors,nodes_IP$Betweenness)
[1] 0.725237
> cor(nodes_IP$Neighbors,nodes_IP$Community)
[1] -0.04173731
> cor(nodes_IP$Neighbors,nodes_IP$Degree)
[1] 0.7750518
> cor(nodes_IP$Closeness,nodes_IP$TotalTransfer)
[1] 0.1128257
> cor(nodes_IP$Closeness,nodes_IP$Betweenness)
[1] 0.1012668
> cor(nodes_IP$Closeness,nodes_IP$Community)
[1] -0.6940921
> cor(nodes_IP$Closeness,nodes_IP$Degree)
[1] 0.0583648
> cor(nodes_IP$Closeness,nodes_IP$Neighbors)
[1] 0.05816097
> cor(nodes_IP$Duration,nodes_IP$TotalTransfer)
[1] 0.3150617
> cor(nodes_IP$Duration,nodes_IP$Betweenness)
[1] 0.265229
> cor(nodes_IP$Duration,nodes_IP$Community)
[1] 0.03160175
> cor(nodes_IP$Duration,nodes_IP$Degree)
[1] 0.3320974
> cor(nodes_IP$Duration,nodes_IP$Neighbors)
[1] 0.6270381
```

Fig. 148. Correlation of Continuous Variables IP Network II

```
> cor(nodes_port$Betweenness,nodes_port$Degree)
[1] 0.9134699
> cor(nodes_port$Betweenness,nodes_port$Community)
[1] -0.008448486
> cor(nodes_port$Betweenness,nodes_port$TotalTransfer)
[1] 0.8971041
> cor(nodes_port$Degree,nodes_port$Community)
[1] -0.01050231
> cor(nodes_port$Degree,nodes_port$TotalTransfer)
[1] 0.8619995
> cor(nodes_port$Neighbors,nodes_port$TotalTransfer)
[1] 0.9566978
> cor(nodes_port$Neighbors,nodes_port$Betweenness)
[1] 0.9428089
> cor(nodes_port$Neighbors,nodes_port$Community)
[1] -0.002463148
> cor(nodes_port$Neighbors,nodes_port$Degree)
[1] 0.9365094
> cor(nodes_port$Closeness,nodes_port$TotalTransfer)
[1] -0.01406633
> cor(nodes_port$Closeness,nodes_port$Betweenness)
[1] 0.02086452
> cor(nodes_port$Closeness,nodes_port$Community)
[1] -0.6436347
> cor(nodes_port$Closeness,nodes_port$Degree)
[1] 0.01426549
> cor(nodes_port$Closeness,nodes_port$Neighbors)
[1] 0.01097084
> cor(nodes_port$Duration,nodes_port$TotalTransfer)
[1] 0.1847325
> cor(nodes_port$Duration,nodes_port$Betweenness)
[1] 0.1834678
> cor(nodes_port$Duration,nodes_port$Community)
[1] 0.2093474
> cor(nodes_port$Duration,nodes_port$Degree)
[1] 0.1964065
> cor(nodes_port$Duration,nodes_port$Neighbors)
[1] 0.1692195
```

Fig. 149. Correlation of Continuous Variables Port Network II

```
> cor(nodes_Pair$Betweenness,nodes_Pair$Degree)
[1] 0.777555
> cor(nodes_Pair$Betweenness,nodes_Pair$Community)
[1] -0.05161002
> cor(nodes_Pair$Betweenness,nodes_Pair$TotalTransfer)
[1] 0.3672247
> cor(nodes_Pair$Degree,nodes_Pair$Community)
[1] -0.08452845
> cor(nodes_Pair$Degree,nodes_Pair$TotalTransfer)
[1] 0.6723512
> cor(nodes_Pair$Neighbors,nodes_Pair$TotalTransfer)
[1] 0.6723512
> cor(nodes_Pair$Neighbors,nodes_Pair$Betweenness)
[1] 0.777555
> cor(nodes_Pair$Neighbors,nodes_Pair$Community)
[1] -0.08452845
> cor(nodes_Pair$Neighbors,nodes_Pair$Degree)
[1] 1
> cor(nodes_Pair$Closeness,nodes_Pair$TotalTransfer)
[1] -0.05922993
> cor(nodes_Pair$Closeness,nodes_Pair$Betweenness)
[1] 0.104757
> cor(nodes_Pair$Closeness,nodes_Pair$Community)
[1] -0.5293187
> cor(nodes_Pair$Closeness,nodes_Pair$Degree)
[1] 0.08975622
> cor(nodes_Pair$Closeness,nodes_Pair$Neighbors)
[1] 0.08975622
> cor(nodes_Pair$Duration,nodes_Pair$TotalTransfer)
[1] 0.1403925
> cor(nodes_Pair$Duration,nodes_Pair$Betweenness)
[1] 0.03892384
> cor(nodes_Pair$Duration,nodes_Pair$Community)
[1] 0.004057489
> cor(nodes_Pair$Duration,nodes_Pair$Degree)
[1] 0.1855039
> cor(nodes_Pair$Duration,nodes_Pair$Neighbors)
[1] 0.1855039
```

Fig. 150. Correlation of Continuous Variables Pair Network II

```
> cor(nodes_PairToIP$Betweenness,nodes_PairToIP$Degree)
[1] 0.6650763
> cor(nodes_PairToIP$Betweenness,nodes_PairToIP$Community)
[1] -0.06780669
> cor(nodes_PairToIP$Degree,nodes_PairToIP$Community)
[1] -0.06979672
> cor(nodes_PairToIP$Neighbors,nodes_PairToIP$Betweenness)
[1] 0.6650763
> cor(nodes_PairToIP$Neighbors,nodes_PairToIP$Community)
[1] -0.06979672
> cor(nodes_PairToIP$Neighbors,nodes_PairToIP$Degree)
[1] 1
> cor(nodes_PairToIP$Closeness,nodes_PairToIP$Betweenness)
[1] 0.1398377
> cor(nodes_PairToIP$Closeness,nodes_PairToIP$Community)
[1] -0.5219617
> cor(nodes_PairToIP$Closeness,nodes_PairToIP$Degree)
[1] 0.04733989
> cor(nodes_PairToIP$Closeness,nodes_PairToIP$Neighbors)
[1] 0.04733989
```

Fig. 151. Correlation of Continuous Variables PairToIP Network II

```
> cor(nodes_PairToPort$Betweenness,nodes_PairToPort$Degree)
[1] 0.7030698
> cor(nodes_PairToPort$Betweenness,nodes_PairToPort$Community)
[1] -0.06016542
> cor(nodes_PairToPort$Degree,nodes_PairToPort$Community)
[1] -0.02569204
> cor(nodes_PairToPort$Neighbors,nodes_PairToPort$Betweenness)
[1] 0.7030698
> cor(nodes_PairToPort$Neighbors,nodes_PairToPort$Community)
[1] -0.02569204
> cor(nodes_PairToPort$Neighbors,nodes_PairToPort$Degree)
[1] 1
> cor(nodes_PairToPort$Closeness,nodes_PairToPort$Betweenness)
[1] 0.1495646
> cor(nodes_PairToPort$Closeness,nodes_PairToPort$Community)
[1] -0.4710188
> cor(nodes_PairToPort$Closeness,nodes_PairToPort$Degree)
[1] 0.03718483
> cor(nodes_PairToPort$Closeness,nodes_PairToPort$Neighbors)
[1] 0.03718483
```

Fig. 152. Correlation of Continuous Variables PairToPort Network II

The centrality plots (Figure 153, Figure 154, Figure 155, Figure 155 and Figure 156) illustrate in parallel the measures of degree, closeness and betweenness and it is very useful to identify outliers or similar behavior within the network behavior. As outlined in the degree histogram, it is also visible in the centrality plot of all network types, that most of the nodes have rather lower degrees, while three nodes have a degree definitely higher than the average. The closeness distribution along all nodes is irregular (proposition 19), although one single node has a higher closeness than the average closeness. This particular node also has the highest betweenness and is among the nodes having a high degree. The betweenness centrality has a similar pattern with degree: many nodes with a low betweenness and a few nodes with a high betweenness. The nodes having a high betweenness are also the nodes having a high degree (proposition 20).

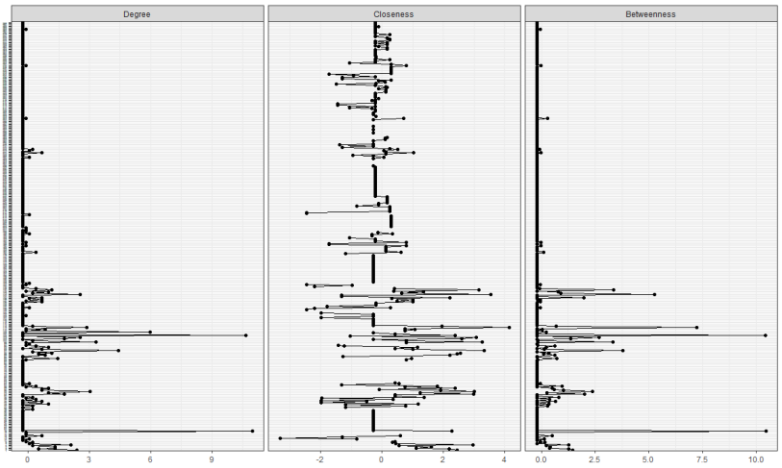

Fig. 153. Centrality Plot IP Network II

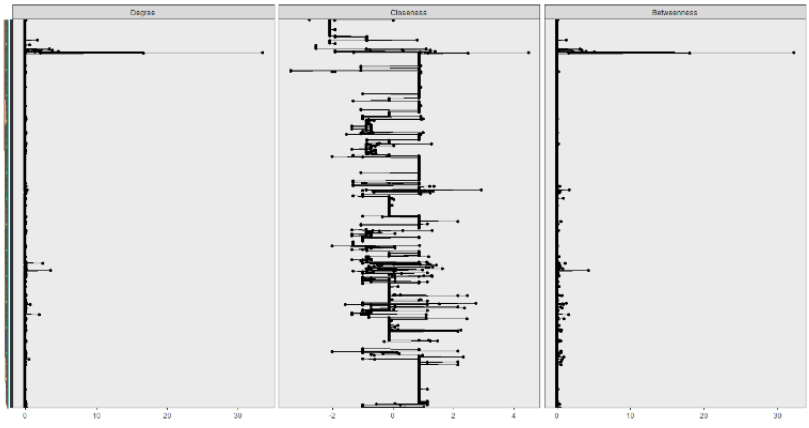

Fig. 154. Centrality Plot Port Network II

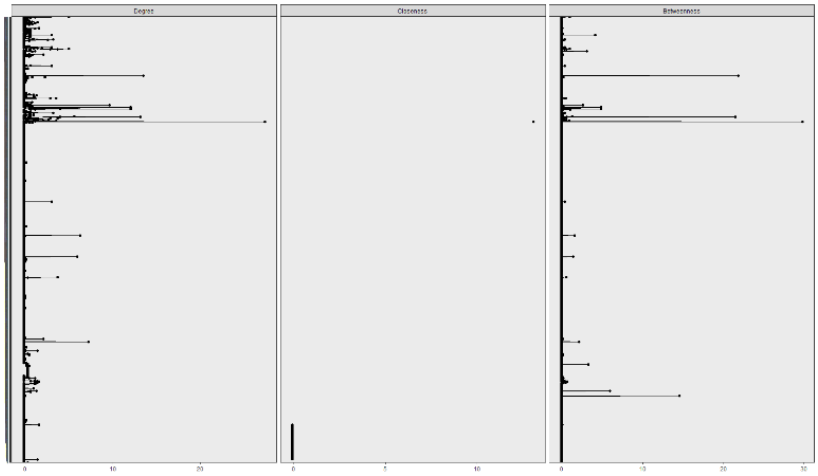

Fig. 155. Centrality Plot Pair Network II

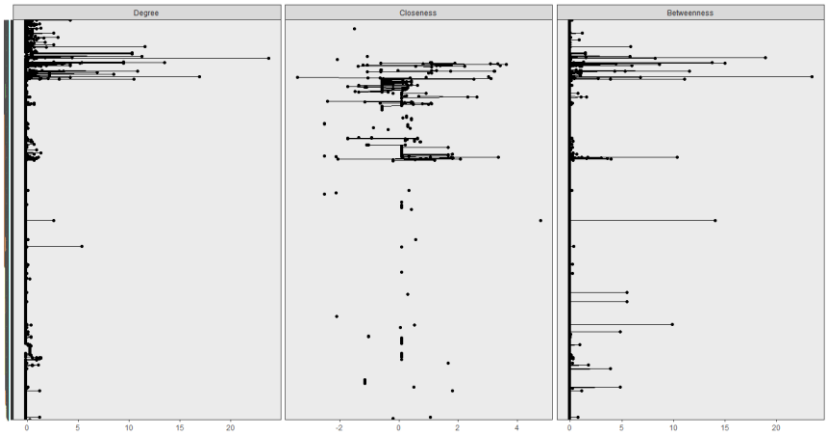

Fig. 156. Centrality Plot PairToIP Network II

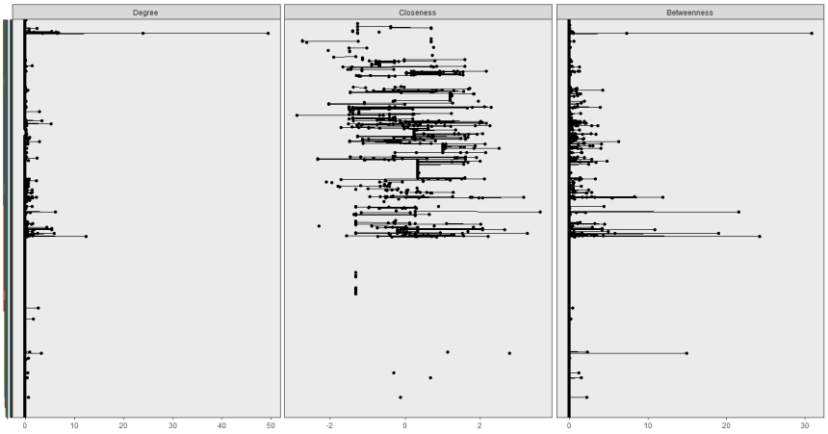

Fig. 157. Centrality Plot PairToPort Network II

The P2P approach identifies approximately 86% of the total P2P flows detected by the original P2P algorithm (proposition 23) and the ports involved in the P2P traffic that send the most amount of bytes are unassigned ports. The hosts assigned to these ports should be monitored.

There are no exactly structural equivalent hosts with more than 10 neighbors in Network II.

Figure 158 describes the density of each equivalent block in terms of an approximate structural equivalence.

```
Network Blockmodel:

Block membership:

  1   2   3   4   5   6   7   8   9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26
  1   3   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   2   3   3   3   3   3   3   3   3
 27  28  29  30  31  32  33  34  35  36  37  38  39  40  41  42  43  44  45  46  47  48  49  50  51  52
  3   3   3   3   3   3   3   3   3   3   3   1   1   1   1   1   1   1   1   1   1   1   3   1   1   1
 53  54  55  56  57  58  59  60  61  62  63  64  65  66  67  68  69  70  71  72  73  74  75  76  77  78
  1   1   1   1   1   1   4   1   1   1   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2
 79  80  81  82  83  84  85  86  87  88  89  90  91  92  93  94  95  96  97  98  99 100 101 102 103 104
  2   2   2   2   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   3   1   1   1
105 106 107 108 109 110 111 112 113 114 115 116 117 118 119 120 121 122 123 124 125 126 127 128 129 130
  3   1   3   3   1   1   1   3   3   3   3   3   3   3   3   3   1   3   1   3   1   1   1   1   1   1
131 132 133 134 135 136 137 138 139 140 141 142 143 144 145 146 147 148 149 150 151 152 153 154 155 156
  1   1   1   1   1   1   1   1   4   1   1   1   1   1   1   4   1   1   1   1   1   1   3   3   3   3
157 158 159 160 161 162 163 164 165 166 167 168 169 170 171 172 173 174 175 176 177 178 179 180 181 182
  3   3   3   3   3   3   3   3   3   3   3   3   3   3   3   3   3   3   3   3   3   1   1   1   1   1
183 184 185 186 187 188 189 190 191 192 193 194 195 196 197 198 199 200 201 202 203 204 205 206 207 208
  1   1   1   4   4   4   4   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1
209 210 211 212 213 214 215 216 217 218 219 220 221 222 223 224 225 226 227 228 229 230 231 232 233 234
  1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   4   4   4   4
235 236 237 238 239 240 241 242 243 244 245 246 247 248 249 250 251 252 253 254 255 256 257 258 259 260
  4   4   4   4   4   4   4   4   4   4   4   4   4   4   4   4   4   4   4   4   4   4   4   3   2   2
261 262 263 264 265 266 267 268 269 270 271 272 273 274 275 276 277 278 279 280 281 282 283 284 285 286
  2   2   2   2   3   1   3   1   1   1   1   1   1   1   1   3   1   3   2   2   3   1   1   1   2   2
287 288 289 290 291 292 293 294 295 296 297 298 299 300 301 302 303 304 305 306 307 308 309 310 311 312
  2   3   2   2   3   2   2   3   2   1   1   2   2   3   3   2   1   2   3   3   2   4   4   1   1   4
313 314 315 316 317 318 319 320 321 322 323 324 325 326 327 328 329 330 331 332 333 334 335 336 337 338
  1   1   4   4   1   4   1   4   4   4   4   4   1   1   1   1   1   1   4   1   1   1   1   1   1   1
339 340 341 342 343 344 345 346 347 348 349 350 351 352 353 354 355 356 357 358 359 360 361 362 363 364
  4   1   4   1   1   1   1   1   1   1   4   1   1   4   4   1   4   1   1   4   4   4   4   4   4   1
365 366 367 368 369 370 371 372 373 374 375 376 377 378 379 380 381 382 383 384 385 386 387 388
  1   1   4   1   4   1   4   1   1   1   4   4   1   4   4   4   4   4   4   1   4   4   4   4

Reduced form blockmodel:

          1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32 33 34 35 36 37 38 39 40 41 42 43 44 45
 46 47 48 49 50 51 52 53 54 55 56 57 58 59 60 61 62 63 64 65 66 67 68 69 70 71 72 73 74 75 76 77 78 79 80 81 82 83 84 85 86 87 88 89 9
0 91 92 93 94 95 96 97 98 99 100 101 102 103 104 105 106 107 108 109 110 111 112 113 114 115 116 117 118 119 120 121 122 123 124 125 1
26 127 128 129 130 131 132 133 134 135 136 137 138 139 140 141 142 143 144 145 146 147 148 149 150 151 152 153 154 155 156 157 158 159
 160 161 162 163 164 165 166 167 168 169 170 171 172 173 174 175 176 177 178 179 180 181 182 183 184 185 186 187 188 189 190 191 192 1
93 194 195 196 197 198 199 200 201 202 203 204 205 206 207 208 209 210 211 212 213 214 215 216 217 218 219 220 221 222 223 224 225 226
 227 228 229 230 231 232 233 234 235 236 237 238 239 240 241 242 243 244 245 246 247 248 249 250 251 252 253 254 255 256 257 258 259 2
60 261 262 263 264 265 266 267 268 269 270 271 272 273 274 275 276 277 278 279 280 281 282 283 284 285 286 287 288 289 290 291 292 293
 294 295 296 297 298 299 300 301 302 303 304 305 306 307 308 309 310 311 312 313 314 315 316 317 318 319 320 321 322 323 324 325 326 3
27 328 329 330 331 332 333 334 335 336 337 338 339 340 341 342 343 344 345 346 347 348 349 350 351 352 353 354 355 356 357 358 359 360
 361 362 363 364 365 366 367 368 369 370 371 372 373 374 375 376 377 378 379 380 381 382 383 384 385 386 387 388
            Block 1      Block 2     Block 3     Block 4
Block 1 0.0135526115 0.0001196458 0.002240934 0.001789771
Block 2 0.0001196458 0.0000000000 0.036357786 0.000000000
Block 3 0.0022409344 0.0363577864 0.000000000 0.012958164
Block 4 0.0017897708 0.0000000000 0.012958164 0.000000000
```

Fig. 158. Equivalence Blocks of IP Network II

The representation of the block model in Figure 159 shows that most of the comparably dense blocks are again outside the diagonal. These are marginal roles (blocks 2, 3 and 4) that are connected to the center (block 1).

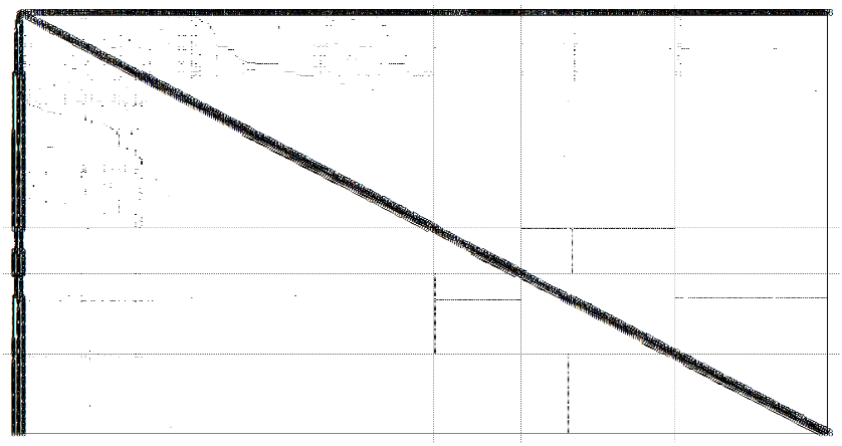

Fig. 159. Representation of Approximately Equivalent Blocks IP Network II

The blocks can be visualized as graph in Figure 161, where the core block is labeled with the color orange. It is visible that all nodes within the P2P cluster represents a structural equivalent block.

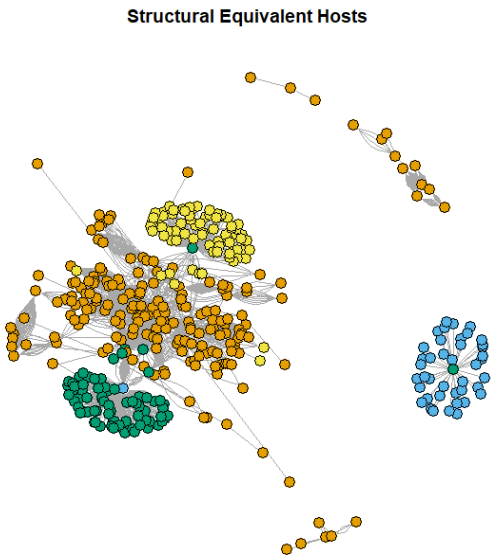

Fig. 160. Visualization of the Structural Equivalence Blocks of Network II

The time plots over the 4 dimensions: transfer in bytes, degree, betweeenness and neighbors reveal again irregularity within the network behavior. It is visible in Figure 161 that the highest amount of bytes are transferred at the beginning (around 16:22) of the data log (proposition 22). On the contrary, the rest of the time interval contains considerably low bytes transfers.

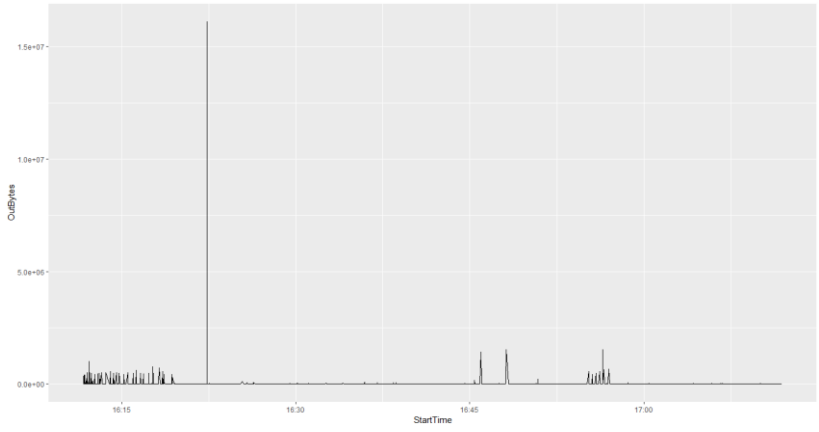

Fig. 161. Bytes Transfer versus Time IP Network II

The degree distribution over time is characterized by an irregular dispersion during the whole time interval (Fig.162). Nevertheless, the degree tends to be either very high or around the degree median.

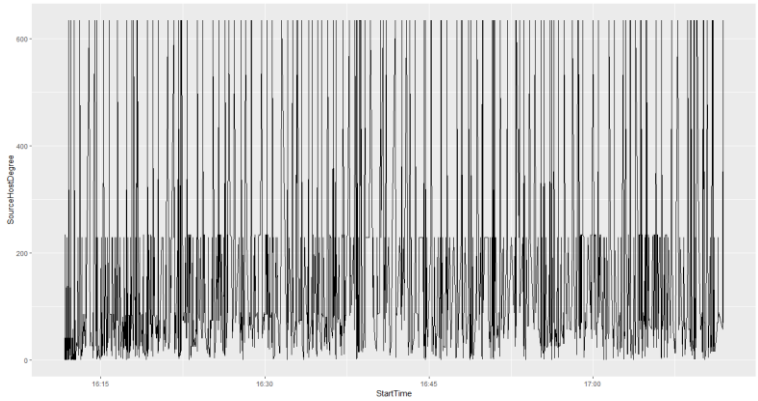

Fig. 162. Degree versus Time IP Network II

The neighbors distribution over time indicates a similar pattern as degree (Figure 163).

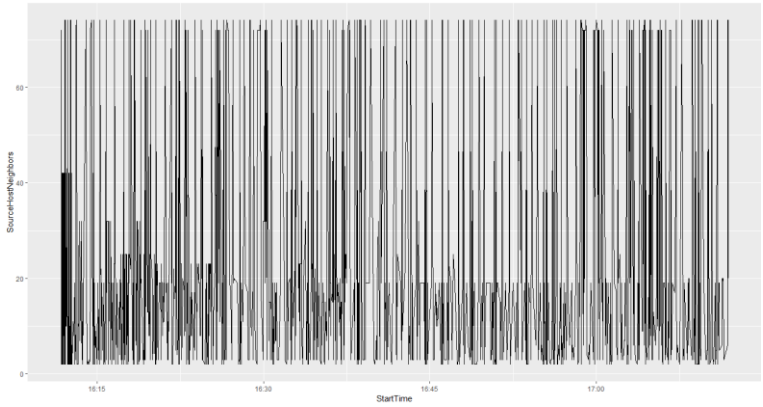

Fig. 163. Neighbors versus Time IP Network II

The betweenness distribution over time (Figure 164) indicates a similar behavior with degree, which is to be expected according to the Pearson correlation of 0,67 between degree and betweenness (Figure 148).

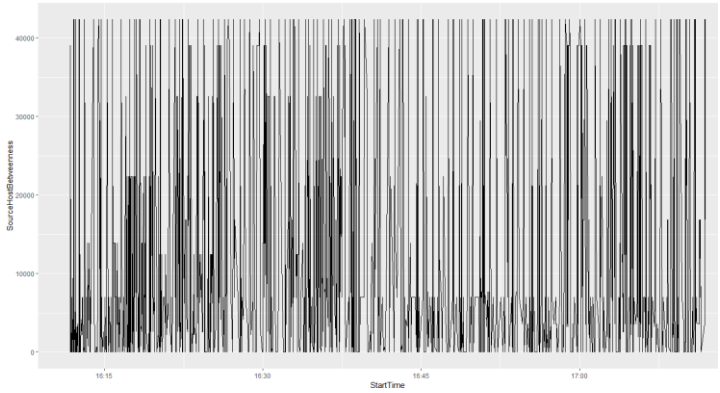

Fig. 164. Betweenness versus Time IP Network II

The time plots of Network II make it difficult to find patterns, respectively outliers and suspicious activity, since the time interval extends over one hour and the plot tends to appear unclear or busy.

22 out of 23 propositions were validated by the analysis of Network 2. The proposition *"21. The core structural equivalence block contains at least one egocentric subnetwork"* cannot be validated, due to the lack of visual clarity in the graph corresponding to Network II.

7 Discussion

Section 1 outlined the motivation including the problem statement and described the research question. Section 2 presented the related work and the corresponding deficiencies in the status of the research. Section 3 defined the chosen methodology that was further applied in both Section 4 and Section 6, which contain the explorative analysis of the data, respectively the evaluation based on a second data set. Section 5 summarized the outcome of the methods applied in Section 4, with the intent of validating these results against the second data set. Getting an overview of the network patterns and its possible anomalies is possible by following the setup instructions described in Section 3.3 and then executing the script with the intent of both visualizing the resulting data frames and the plots within the R environment and of automatically generating a HTML report. All these outcomes lead to the answering of the research questions.

RQ) What is the extent to which the measures of SNA can be used to find communication patterns within the network and what is the meaning of these patterns?

The results of the methodology used lead to the understanding of the structural and behavioral characteristics of an enterprise network and they are summarized in form

of the conceptual propositions formulated in Section 5 and 22 out of 23 being validated in Section 6. Following observations outline the meaning of the communication patterns found:

- Hosts that use a single source port for the majority of their interactions are likely to be providers of a service offered on that port.
- Assets that have more ties (higher degrees) and egos who bridge gaps between the vertices may be advantaged and have fewer constraints, since they have choices. This autonomy grants them an independence regarding other actors and makes them more powerful [13, p.146].
- The hosts having high values of centrality values highlight the way traffic flows, by suggesting the speed of information transfer and receiving frequency.
- A high average degree of a node in the IP Network indicates a high user variability [40, p.196], which is common in mixt and enterprise networks.
- According to [42, p.1176], a node linked to linked nodes is central, but not powerful, since the linked nodes are not dependent on it.
- The centrality measures (betweennes, degree, closeness) produce expected values for specific types of vertices outcomes (such as speed and regularity of reception) [38, p.56].
- The egocentric subgraphs highlight the most influential hosts.
- Actors who have more ties (higher degrees) and egos who bridge gaps between the vertices are advantaged. Because they have many links, they have access to more network resources and collaborate in transfers with other nodes and can profit from this brokerage [13, p.147].
- The more links a node has to another node, the greater the probability that communication takes place, and the safer the linkage [13 ,p.113]
- High betweenness values reveal which node plays a key role in facilitating the direct flow of information between itself and other nodes within the network. The node having a high betweenness has “the capacity to broker contacts among other actors (…) and to isolate actors or prevent contacts” [13, p.147] .
- A highly disconnected graph, respectively network, can generate undesired consequences, since it is hardly manageable and is predisposed to attacks.

RQ/SQ1) Which communication patterns within a network of interest are determined by the given attributes of the network?

The anomalous traffic is detected by using the port numbers, the protocol and the transfer bytes/packets, as described in Section 3.5. The identification of communication patterns of common services (web traffic, DNS, mail, remote, VPN) imply the usage of the port numbers and the protocol, as outlined in Section 3.6. Since the recognition of P2P activity implies the consideration of additional patterns, such as scans, DNS, port history and gaming/malware, the P2P traffic pattern is identified by

using the IPs, the ports, the protocol and the transfer bytes and packets. These patterns are visually represented in Section 3.8.

RQ/SQ2) Which network assets require a closer investigation based on the results of the applied methodology?

The measures of SNA provide a broad overview of the network in terms of its structure and behavior and offer an insight on which network assets need to be monitored, since they may generate anomalous activity or P2P traffic. Following observations provide a guidance for the next steps that should be taken to increase the enterprise network security and robustness:

- Since a cohesive network is both less predisposed to external attacks and easier to maintain, it is highly recommended to connect the clusters forming the network by installing additional assets that link these clusters.
- A rigorous monitoring of the smaller subgraphs of the network is recommended, since at least one of them should contain exclusively P2P traffic, which is insecure.
- If a higher speed or regularity of reception is desired, then the assets having a low value of betweenness or degree should be investigated and optimized.
- Since a high number of flows between two adjacent assets provides a higher probability that communication takes place, and that this communication is less vulnerable, it is highly recommended to monitor and optimize the assets having a few links assigned to them.
- All articulation points should be monitored closely, since their removal disconnects the network.
- If a client has a similar behavior to a server, this could mean that the client is compromised or violates the security policies [39, p.53]. Thus, the assets assigned to these clients should be investigated.
- Supervise hosts that have a maximum value of betweenness (the centers of an egocentric network), since these hosts are central, hence "powerful" within their neighborhood [13, p.135]. Thus, their neighboring hosts depend on them. According to [13, p.144], "ego's power is alter's dependence".
- The hosts assigned to the P2P ports responsible for the largest amount of P2P traffic within the network should be monitored.

Although the methodology used is straightforward, there are still some challenges and issues that need to be considered when analyzing the network. These challenges and the algorithm's features and limitations are discussed in Section 7.1 and Section 7.2.

7.1 Data Pre-Processing Challenges

The initial files may have missing data, for instance the type of service or the interface number. This results in false positives when profiling the anomalous behavior of the network. For instance, the second data set does not contain any value for the column of the type of service. This results in identifying all flows as "invalid_ToS" and even if this outcome does not affect in any manner the further results, this issue should be considered when firstly processing the data.

The data preparation in Wireshark assumes that each flow contains only one transfer packet. Nevertheless, the data export from Wireshark usually contains a null column corresponding to the packets, so the user must insert manually the value 1 in the CSV file before importing it into the R Environment.

7.2 Limitations

- Dealing with NATs (Network Address Translation): NAT should not influence the profiling as long as the servers are not behind the NAT. A challenge represents also the hosts behind NAT that use at least one service with control and data streams at the same time (for instance FTP).
- Handling the Dynamic Host Configuration Protocol (DHCP). It is assumed that all IP addresses in the data sets are unique host identifiers, although it might be the case that these addresses are mapped to the MAC (media access control address) from the DHCP logs. If this is the case, the user must first assure that all the flow records of each unique IP address are defined by a unique attribute.
- The script removes in the first step all unidirectional flows having no response edge, but it might be the case that a particular response edge is captured outside the time interval of the data frame.
- Worms, Trojans and P2P can use additional ports than the ones hardcoded in the script, so an idea would be that the user updates the ports within the script as soon as a new port presents interest for the research.
- The time and the supplies necessary to run this process rely upon the network size and how well the services are harmonized with the common practices.
- Considering that suspect traffic cannot be always differentiated by normal traffic, the removing of unwanted traffic in the pre-processing step is not 100% accurate.
- The absence of the TCP flag does not guarantee that the 3-way handshake actually occurred.
- Several P2P protocols use HTTP requests and responses to transfer files, and it can be impossible to distinguish such P2P traffic from typical web traffic.

– Since large networks contain a lot of information, they are difficult to visualize and to find patterns within them. An alternative solution is to import these plots in Gephi, for instance.

8 Summary and Outlook

This thesis provides an overview of a network behavior by using a methodology based on both an extension of the traditional methods of traffic profiling and SNA. The overview is given by several visualizations, statistical results and data frames, which can further support a security architect or a network engineer in a better understanding and management of the network of interest. The statistical results are mapped on flow level in the main data frame and summarized in form of a HTML report. A summary of the communication patterns and their characteristics can be visualized in the appendix of this thesis (Section A, Table 1).

One of the goals of this thesis is to generate a profile describing how the network is seen from the outside including an attacker's perspective. Another goal is to use network flow data so that users can find and analyze various assets and services of interest that are responsible for the majority of the communication within the network.

The underlying notions of SNA are outlined and correlated with the initial profiling of the network, so that the users can identify the main servers, potential threats and P2P traffic by only analyzing the network dynamics. A user applying the outlined methodology to profile a network gets an overview of the most exposed assets including their assigned ports, the possible anomalies and the P2P traffic, all mapped both on node and on flow level.

Using R as both an analysis and visualization tool can increase the productivity of work of network managers, by making the analysis of the network flow data approachable also by non-specialists. Other visualization tools such as NFSen or NFDump require their users to write their own commands or configure filters [41, p.150].

Although the HTML report provides a broad overview of the network behavior and structure by outlining the key data frames and the visualizations resulted from the analysis, it is necessary that the user carries a basic theoretical knowledge of R programming in order to get a picture of the whole dimension of the data frames but also to modify the script, if required. A large amount of the computational effort needed for data pre-processing in R is undertaken by Wireshark, which provides the columns needed for the analysis and the right data format.

The data needs to be uploaded into the R environment and the script needs to be executed so that the data can be cleaned and filtered for the analysis and the user can get an accurate overview of the network characteristics and its several visualizations. The R script is built around the applied methodology and it presents the convenience that it can be modified by the user to fulfill additional requirements. The script is supported by comments for each process phase, allowing the user to know at each step

how the outcome of the code chunk is created. The visualizations cannot replace other tools created mainly for this purpose but it is also expected that these other tools do not provide analysis algorithms and identification of common services or suspicious activity. The chosen visualizations serve to show broadly the network behavior and structure and some of them may appear as noise to the user, as a result of overlapping nodes and not identifiable elements. If this is the case, then the underlying data frames need to be inspected, since they all serve as basis for the visualizations.

The data frames and the plots in the R environment together with the HTML report provide ideas for optimizing and securing the network. The HTML report has to be used together with the data frames within the R environment and with the findings outlined in this thesis for a comprehensive understanding of the network behavior and structure. A user can further analyze in detail the assets of the network if the traffic analysis outcomes seem interesting or out of place. The analysis is also helpful for the evaluation of the assets that violate policies and for detecting any suspicious activity.

The conceptual propositions in Section 5 and the results in Table 1 can be generalized for other data sets besides the AIT and LBNL data. Following conditions should be fulfilled in order to be able to generalize these findings. The data sets should represent an internal enterprise traffic and the IP addresses must be unique host identifiers. To avoid misleading results, the traffic should be captured within working hours. If the export traffic files are in a different format than PCAP/NFCAPD, they must first be converted to one of these formats. Also, the 11 mandatory columns (include the start- and end time, the source and destination address , the source and destination port, the protocol, the source type of service, the input packets, the input bytes and the input interface number) should be included in the data sets and should contain values.

The implementation of the chosen methodology revealed some weak points with regard to the visualization techniques and to the scalability of the input data sets. Following ideas provide an insight on how future efforts can be made in order to improve these deficiencies, respectively on how to address in greater detail the research question:

- An online anomaly detection and network profiling system based on the chosen methodology would improve the user interaction design.
- It would be meaningful to analyse the network traces in terms of organizational mining, which results by applying social network analysis methods on process logs [25, p.290]. For this purpose, several instances of the network traffic are necessary (for example, capturing the network traffic daily at the same time of the day, one weeklong). This approach would be useful in order to assess the similarity between the network actors and the frequency with which a particular network asset (e.g. an IP address) performs a specific task (e.g. usage of the UDP protocol).
- The usage of trace-driven traffic simulations would help to evaluate the false positives and false negatives of this methodology.

- Since Excel cannot handle more than 1.048.576 rows, it is impossible to examine large data sets at one time, so the data needs to be split into several chunks that are then individually analyzed. An alternative solution would be to use Power BI[35], which is a software that can handle large data and it also makes possible to execute a R script on the data.
- The plots in R may be sometimes "busy" due to the large amount of nodes and/or edges, making the labels appear unclear. A solution is to import the particular plot into a software that makes possible an appropriate visualization of large networks.
- Since the information contained in the application payload level is useful for validating the chosen methodology that is based on flow-level header information, it is meaningful to extend this approach by investigating also the packet payload.
- Consider also the presence of NATs, since hosts behind NATs use at least one service with control and data streams at the same time.

[35] https://powerbi.microsoft.com/en-us/

References

1. Ren, M., Jiang, Y., Guo, X., Han, Q., Wen, H., Wu, B., & Chen, Z. (2014, October). P2P networks monitoring based on the SNA and the topological potential. In *Communications and Network Security (CNS), 2014 IEEE Conference on* (pp. 492-493). IEEE.
2. Albdair, M., Addie, R. G., & Fatseas, D. (2016, December). Inference of social network behavior from Internet traffic traces. In *2016 26th International Telecommunication Networks and Applications Conference (ITNAC)* (pp. 7-12). IEEE.
3. Jakalan, A., Gong, J., Su, Q., Hu, X., & Abdelgder, A. M. (2016). Social relationship discovery of IP addresses in the managed IP networks by observing traffic at network boundary. *Computer Networks*, *100*, 12-27.
4. Whisnant, A., & Faber, S. (2012). *Network profiling using flow*(No. CMU/SEI-2012-TR-006). CARNEGIE-MELLON UNIV PITTSBURGH PA SOFTWARE ENGINEERING INST.
5. Karagiannis, T., Broido, A., & Faloutsos, M. (2004, October). Transport layer identification of P2P traffic. In *Proceedings of the 4th ACM SIGCOMM conference on Internet measurement*(pp. 121-134). ACM.
6. Kandula, S., Chandra, R., & Katabi, D. (2008, August). What's going on?: learning communication rules in edge networks. In *ACM SIGCOMM Computer Communication Review* (Vol. 38, No. 4, pp. 87-98). ACM.
7. Iliofotou, M., Pappu, P., Faloutsos, M., Mitzenmacher, M., Singh, S., & Varghese, G. (2007, October). Network monitoring using traffic dispersion graphs (tdgs). In *Proceedings of the 7th ACM SIGCOMM conference on Internet measurement* (pp. 315-320). ACM.
8. Xu, K., Wang, F., & Gu, L. (2011, April). Network-aware behavior clustering of Internet end hosts. In *INFOCOM, 2011 Proceedings IEEE* (pp. 2078-2086). IEEE.
9. Shang, R., Luo, S., Li, Y., Jiao, L., & Stolkin, R. (2015). Large-scale community detection based on node membership grade and sub-communities integration. Physica A: Statistical Mechanics and its Applications, 428, 279-294.
10. Jakalan, A., Gong, J., Weiwei, Z., & Su, Q. (2015). Clustering and profiling ip hosts based on traffic behavior. Journal of Networks, 10(2), 99.
11. Xu, K., Zhang, Z. L., & Bhattacharyya, S. (2005, August). Profiling internet backbone traffic: behavior models and applications. In ACM SIGCOMM Computer Communication Review (Vol. 35, No. 4, pp. 169-180). ACM.
12. Meiss, M., Menczer, F., & Vespignani, A. (2011). Properties and evolution of Internet traffic networks from anonymized flow data. *ACM Transactions on Internet Technology (TOIT)*, *10*(4), 15.
13. Hanneman, R. A., & Riddle, M. (2005). Introduction to social network methods.
14. Kinable, J. (2008, June). Detection of network scan attacks using flow data. In *9th Twente Student Conference on IT, 23th June*.

15. Bo, X., Ming, C., Fei, L., & Na, W. (2009, October). P2P flows identification method based on listening port. In *Broadband Network & Multimedia Technology, 2009. IC-BNMT'09. 2nd IEEE International Conference on* (pp. 296-300). IEEE.
16. Tanenbaum, A. S., & Wetherall, D. (1996). Computer networks (ed.). *ISBN: 0-12-066102-3*.
17. Dordal, P. (2017). An introduction to computer networks.
18. Xu, K., Zhang, Z. L., & Bhattacharyya, S. (2008). Internet traffic behavior profiling for network security monitoring. *IEEE/ACM Transactions on Networking (TON)*, *16*(6), 1241-1252.
19. Sperotto, A., Schaffrath, G., Sadre, R., Morariu, C., Pras, A., & Stiller, B. (2010). An Overview of IP Flow-based Intrusion Detection. *IEEE Communications Surveys and Tutorials*, *12*(3), 343-356.
20. Aiello, W., Kalmanek, C., McDaniel, P., Sen, S., Spatscheck, O., & Van der Merwe, J. (2005, March). Analysis of communities of interest in data networks. In *International Workshop on Passive and Active Network Measurement* (pp. 83-96). Springer, Berlin, Heidelberg.
21. Burgers, B. (2007). Evolvements of TCP-, UDP-, short-lived TCP-and long-lived TCP flows.
22. Zoho Corporation (2018). *Security Problem & Class Catalog*. Retrieved from https://www.manageengine.com/products/netflow/problem-class-catalogue.html
23. Bin, L., Chuang, L., Jian, Q., Jianping, H., & Ungsunan, P. (2008). A NetFlow based flow analysis and monitoring system in enterprise networks. *Computer Networks*, *52*(5), 1074-1092.
24. Erbschloe, M. (2004). *Trojans, worms, and spyware: a computer security professional's guide to malicious code*. Elsevier.
25. Grossmann, W., & Rinderle-Ma, S. (2015). Fundamentals of business intelligence. Springer.
26. Seid, H. A., & Lespagnol, A. (1998). *U.S. Patent No. 5,768,271*. Washington, DC: U.S. Patent and Trademark Office.
27. Bo, X., Ming, C., Fei, L., & Na, W. (2009, October). P2P flows identification method based on listening port. In *Broadband Network & Multimedia Technology, 2009. IC-BNMT'09. 2nd IEEE International Conference on* (pp. 296-300). IEEE.
28. Fraleigh, C., Moon, S., Lyles, B., Cotton, C., Khan, M., Moll, D., ... & Diot, S. C. (2003). Packet-level traffic measurements from the Sprint IP backbone. *IEEE network*, *17*(6), 6-16.
29. Martino, F., & Spoto, A. (2006). SNA: A brief theoretical review and further perspectives in the study of Information Technology. *PsychNology Journal*, *4*(1), 53-86.
30. Kolaczyk, E. D., & Csárdi, G. (2014). *Statistical analysis of network data with R* (Vol. 65). New York: Springer.
31. Karagiannis, T., Papagiannaki, K., & Faloutsos, M. (2005, August). BLINC: multilevel traffic classification in the dark. In *ACM SIGCOMM computer communication review* (Vol. 35, No. 4, pp. 229-240). ACM.

32. Dewaele, G., Himura, Y., Borgnat, P., Fukuda, K., Abry, P., Michel, O., ... & Esaki, H. (2010). Unsupervised host behavior classification from connection patterns. *International Journal of Network Management*, *20*(5), 317-337.
33. Scott, J., & Carrington, P. J. (2011). *The SAGE handbook of SNA*. SAGE publications.
34. Newman, M. E., & Girvan, M. (2004). Finding and evaluating community structure in networks. *Physical review E*, *69*(2), 026113.
35. Clauset, A., Newman, M. E., & Moore, C. (2004). Finding community structure in very large networks. *Physical review E*, *70*(6), 066111.
36. Ligh, M. (2003, August). *Attack Signatures and Internet Traffic Analysis*. Retrieved from https://www.mnin.org/write/2003_asita.html
37. Iliofotou, M., Gallagher, B., Eliassi-Rad, T., Xie, G., & Faloutsos, M. (2010, November). Profiling-By-Association: a resilient traffic profiling solution for the internet backbone. In *Proceedings of the 6th International COnference* (p. 2). ACM.
38. Borgatti, S. P. (2005). Centrality and network flow. *Social networks*, *27*(1), 55-71.
39. Li, B., Gunes, M. H., Bebis, G., & Springer, J. (2013, June). A supervised machine learning approach to classify host roles on line using sFlow. In *Proceedings of the first edition workshop on High performance and programmable networking*(pp. 53-60). ACM.
40. Uhlig, S., Papagiannaki, K., & Bonaventure, O. (2007). Passive and Active Network Measurement. In *8th Internatinoal Conference, PAM 2007, Louvain-la-neuve, Belgium, April 5-6, 2007, Proceedings*.
41. Minarik, P., & Dymacek, T. (2008). Netflow data visualization based on graphs. In *Visualization for computer security* (pp. 144-151). Springer, Berlin, Heidelberg.
42. Bonacich, P. (1987). Power and centrality: A family of measures. *American journal of sociology*, *92*(5), 1170-1182.
43. Sabidussi, G. (1966). The centrality index of a graph. Psychometrika, 31(4), 581-603.
44. Kumar, P. (2011, March 24). Detecting Suspicious Flows using NetFlow Analyzer [Web log post]. Retrieved August 12, 2018, from https://blogs.manageengine.com/network/netflowanalyzer/2011/03/24/detecting-suspicious-flows-using-netflow-analyzer.html
45. Li, B., Springer, J., Bebis, G., & Gunes, M. H. (2013). A survey of network flow applications. *Journal of Network and Computer Applications*, *36*(2), 567-581.

A Summary of the Communication Patterns

Pattern	Characteristics	Observations
Web Server	- Destination IPs using the destination ports 80, 8080, 443, 81, 82,8090. - TCP protocol - Placed within one single network component or within one single cluster. -It can be assigned with a high probability to an egocentric subgraph having a large amount of neighbors.	- Connections with long-duration and a high transfer on port 443 can represent SSL VPN traffic. - Web servers with many client connections can be gateways. - Already identified web servers using the ports 1935, 1755 and 554 provide a streaming media service.
Web Client	- Source IPs using the destination ports 80, 8080, 443, 81, 82,8090. - TCP protocol. - Most of the DNS clients are also web clients	- If there is only one client web traffic to an IP address, then this address should not be considered as a web client. - Any service can be configured to use the ports designated by default for web traffic
Email	- IPs using the destination ports 25, 465, 110, 995, 143 or 993. - Protocol TCP - Email servers are placed within one single network component or within one single cluster. -Email servers can be assigned with a high probability to an egocentric subgraph having a large amount of neighbors.	- An email server can be profiled also as a web server. - Also a client can run an email server, since email services do not necessarily run on dedicated servers. If this is the case, the policies concerning clients running on these services and the firewall configurations need to be checked. - Desktop clients that send SMTP messages should be investigated, i.e., all clients that send messages straight to the internet.
DNS	- IPs using Destination Port 53. - Protocol UDP.	- Server-to-server requests on port 53 are the consequence

	- Transfer of at least 1% of the total bytes. - DNS servers are placed within one single network component or within one single cluster. -DNS servers can be assigned with a high probability to an egocentric subgraph having a large amount of neighbors. -Most of the DNS clients are also web clients	of using an older implementation of DNS. - If there are DNS responses from any host except the ones belonging to the network's DNS servers, this could signify a suspicious activity or a violation of the policy.
VPN	- Protocols ESP, GRE, AH. - Transfer of at least 1% of the total bytes.	- VPN can run also on non-standard ports, which have to be manually configured so that they can have access through the firewall. - Distinct timeouts can be visible across VPN, depending on the vendor's choice. Independent of the timeout length, the patterns within the flows remain the same.
Remote Services	- IPs using the ports 21 (FTP), 22 (SSH), 23 (Telnet)	- Some FTP servers are FTP clients, as well. - Timeouts (no server response), due to the configuration of the firewalls. - Control channel without data channel, due to either the active use of keep-alive packets or to brute force attacks on the FTP servers
Gaming/Malware	- Hosts using the ports addressed in Section 3.5 - All flows containing an (IP, port) pair that have the same average packet size.	
Anomalous Activity	- Identification of Denial of Service, Suspect Flows and Worms/Trojans according to	

	the heuristics outlined in Section 3.5 - Nodes with high degrees	
Scans	- Number of IP in pair / number of IPs connected to IP is above 15	
Port History	- (IP, port) pairs where the ports connected to them are below 1024 and the pair appears in less than 10 flows.	
P2P	- All flows that contain (IP, port) pairs which connect to the same number of IPs and ports. - All flows that contain an (IP, IP) pair using simultaneously TCP and UDP - All flows that contain the P2P ports outlined in Section 3.7 - If the difference between the neighbors of the PairToIP network corresponding to the source pair and the neighbors of the PairToPort network corresponding to the destination pair is greater or equal to zero, the ports are above 1024 and the protocol is TCP/UDP, then more than 85% of the total P2P flows are identified.	- Excluded are DNS, mail, scans, port history, gaming and malware, Netbios, Streaming Media, NTP, ISAKMP, IRC.
Download	- IP addresses having one link to one IP address	
Browsing	- An IP address sends at least one packet to more than two other IP addresses in the trace and to less IP addresses than the threshold value.	
Netscan	- The number of IP addresses connected to one IP address exceeds the threshold value.	

Hosts responsible for disconnecting the network graph	- Articulation points	-Monitor all articulation points
Influential Hosts	- The center of egocentric subgraphs.	
High likelihood of communication occurrence	- Hosts having a medium to high degree value and/or many neighbors	
Nodes playing a key role in facilitating the direct flow of information between itself and other nodes within the network	- High betweenness centrality	
Signaling Traffic	Small-Size packets	
Data Exchange	Large-Size Packets	

Table 1. Summary of the Communication Patterns

B Source Code

B.1 Preparation

The main R script *("SourceCode.R")* can be executed within any version of RStudio equal to 1.1.383 or higher. Following steps are necessary for the preparation:

- Convert the NFCAPD file format or the PCAP file format to CSV format (as summarized in Section 3.3) and save it in a local folder.
- Install any version of RStudio equal or higher than 1.1.383.
- Load the CSV file into the R environment.
- Type *getwd()* within the R console to get the working directory.
- Replace the working directory in *setwd("C:/Users/Lenovo/Desktop/Masterarbeit/")* with the result of the command *getwd()* or choose a new path.
- Replace "rawdataCSV" in *rawdata = readCSV("rawdataCSV", header = TRUE, sep = ";",check.names=FALSE,row.names=NULL)* with the name of the CSV file.

B.2 Usage

Following steps should be followed in this exact order in order to avoid possible errors:

- Firstly, the necessary packages need to be installed and loaded (R script "Packages.R").
- After performing the steps described in Section B.1 Preparation, the main R script is executed, either line by line or the whole script at once.
- The R Markdown report *"SummaryReport.RMD"* is used for generating the HTML report, based on inputs from the main R script. The R script has to be executed prior to the R Markdown report. To minimize the execution time of the R Markdown file, it is recommended to comment the code lines 2018 – 3152 from the main R script when "Knitting" the document.

B.3 Data Frames

All data frames can be found in the RStudio global environment and are generated gradually during the code execution. Following data frames provide valuable information for the purpose of a further in-depth analysis:

- DF11: contains the initial flow table extended by the results of the network profiling.
- nodes_IP: contains the attributes of the IP Network.
- nodes_Pair: contains the attributes of the Pair Network.
- nodes_Port: contains the attributes of the Port Network.
- nodes_PairToIP: contains the attributes of the PairToIP Network.
- nodes_PairToPort: contains the attributes of the PairToPort Network.
- dns_clients: contains the IP and Port of the DNS clients.
- dns_servers: contains the IP and Port of the DNS servers.
- dns_recursive: contains the IP and Port of the DNS recursive.
- webclients: contains the IP and Port of the web clients.
- web_servers: contains the IP and Port of the web servers.
- vpn_servers: contains the IP and Protocol of the VPN servers.
- mail_server: contains the IP and Port of the mail servers.
- mail_clients: contains the IP and Port of the mail clients.
- p2p: contains the source and destination IP of the p2p hosts.
- flow_anomalies: contains the 5-tuple flows identified as suspect flows, denial of service and/or worms/Trojans.

The rest of the data frames support the creation of the data frames enumerated. They contain particular information and their names are self-explanatory, in case the interest exists for accessing this information.

C Abstract

C.1 English

Network flow data is usually gathered for network monitoring and administration, as well as for security analysis. By using commercial or open source tools variations from traditional traffic criterions and communications with acknowledged bad hosts can be identified. While it is recommended to use existing tools to address and solve issues, the maintenance of these tools brings a new layer of complexity because they do not support all network's dimensions that a business needs. Resources are therefore spent on analyzing separate network elements rather than on understanding how the network as a whole needs to be optimized and also secured against intruders. Social Network Analysis uses graph theory for exploring social structures and focuses on the relations and the patterns they form and although it does not provide a set of premises from which hypothesis or predictions can be derived as network analysis tools do, it provides guidance on where to look further for the answers. This thesis provides the first-of-its-kind network structure and behavior analysis based on graph theory. A top-down perspective is adopted, which aims to characterize the whole population by the relations patterns that constrain its actors.

C.2 German

NetFlow Daten werden normalerweise zur Netzwerküberwachung und -verwaltung sowie zur Sicherheitsanalyse erfasst. Durch den Einsatz von kommerziellen oder Open-Source-Tools können sowohl Abweichungen von traditionellen Netzwerkverkehrskriterien, als auch die Kommunikation mit bekannten verdächtigen Hosts identifiziert werden. Es wird zwar empfohlen, vorhandene Tools zur Lösung von Problemen zu verwenden, ihre Wartung bringt jedoch eine neue Komplexitätsebene mit sich. Der Grund hierfür ist, dass sie nicht die netzwerkweite Dimensionalität darstellen, die ein Unternehmen benötigt. Ressourcen werden daher für die Analyse einzelner Netzwerkelemente aufgewendet, anstatt um zu verstehen, wie das Netzwerk als Ganzes optimiert und auch vor Angreifern geschützt werden soll. Die Analyse von sozialen Netzwerken verwendet Graphentheorie um soziale Strukturen zu untersuchen und basiert auf den Beziehungen und ihren zugrundeliegenden Mustern. Obwohl die Analyse von sozialen Netzwerken keine Prämissen vorgibt, von denen Hypothesen oder Vorhersagen abgeleitet werden können, wie Netzwerkanalysetools es tun, bietet sie eine Anleitung an, wonach man weiter nach Antworten suchen kann. Diese Masterarbeit bietet die erste Netzwerkstruktur- und Verhaltensanalyse auf Basis von Graphentheorie. Ein Top-down-Ansatz wird verwendet, die darauf abzielt, die gesamte Grundgesamtheit durch die Kommunikationsmuster zu charakterisieren, die ihre Akteure einschränken.

D Summary – German

Diese Arbeit setzt sich als Ziel, einen neuartigen Ansatz für die Klassifizierung von Internetdiensten, einschließlich der Visualisierung ihrer zugrundeliegenden Verhaltensmuster und der Identifikation von Kommunikationsmustern innerhalb des Netzwerkverkehrs mittels sozialer Netzwerkanalyse, bereitzustellen. Die Ergebnisse der Klassifizierung und der SNA-Methoden werden auf der Datenflussebene abgebildet, so dass die gesamte Dimension des Netzwerks abgedeckt werden kann. Somit können auch Kommunikationsregeln, die die meisten Benutzeraktivitäten regeln, bereitgestellt werden.

Die Literaturrecherche liefert nur ein paar Ansätze für einige Teilbereiche dieser Arbeit und diese Ergebnisse führten zusammen mit den eigenen Ideen zur Abdeckung der gesamten Dimension des Internetverkehrs. Die Analyseergebnisse und die entsprechenden konzeptionellen Aussagen, die durch die Analyse eines Datensatzes erhalten werden, werden validiert, indem die gleichen Methoden auf einen anderen Datensatz angewendet werden.

Der zweite Datensatz weist ähnliche Kommunikationsmuster auf wie der erste Datensatz, selbst wenn die Netzwerkstruktur unterschiedliche Eigenschaften in Bezug auf die Gesamtdauer, den Ort der Datenaufzeichnung und die verwendeten Dienste aufweist. Die Methoden der sozialer Netzwerkanalyse werden über Regeln hinaus angewendet, die nur die IP-Adressen als signifikante Einheiten von Interesse für die Analyse umfassen, und erstrecken sich auf (IP, Port) - Paare-Analyse. Solch ein expansives Bild stellt einem Netzwerkadministrator einen Realitätscheck zur Verfügung. Es kann gesehen werden, wie der Netzwerkverkehr, der auf führende Anwendungen zurückzuführen ist, das Netzwerk prägt. Konfigurationsfehler, fehlerhafte Kommunikation und verdächtige Aktivitäten können soeben erkannt werden. Zu diesem Zweck wird das Ergebnis dieser Analyse in Form eines HTML-Berichts zusammengefasst, der Hinweise darauf gibt, welche Assets des Netzwerks einer genaueren Untersuchung und Überwachung bedürfen. Das für die Analyse verwendete R-Skript kann vom Benutzer manuell konfiguriert werden, um die gewünschten Anforderungen zu erfüllen. Das Skript kann wiederholt auf verschiedene Netzwerkdaten angewendet werden, um Abhängigkeiten für die verwendeten Anwendungen zu lernen. Die Ergebnisse können sowohl für Netzwerkadministratoren als auch für Benutzer interessant sein, die an einem grundlegenden Verständnis eines bestimmten Netzwerkverkehrs interessiert sind.