POLITECNICO DI TORINO

Dipartimento di Elettronica e delle Telecomunicazioni

Master of Science in
Communications and Computer Networks Engineering

Master Thesis in Network Monitoring and Data Analysis

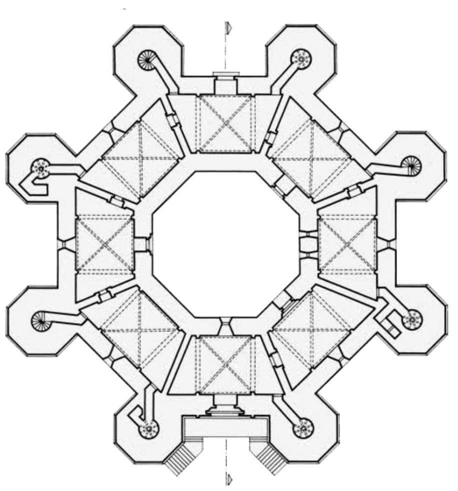
Community Detection Algorithms for Darknet Traffic Analysis

Supervisors:
Prof. Marco Mellia
Dr. Idilio Drago

Candidate:
Mauro Allegretta

October 2019

a mia madre, a mio padre,
a chi mi vuole del bene,
oltre ogni ragionevole dubbio



“ e volta nostra poppa nel mattino
de’ remi facemmo ali *al folle volo ”*

Ulisse, ***Inferno****, XXVI, vv.124-125, Dante*

Abstract

Today the diffusion of internet is widespread and so the defence from **cyber-attacks** is very relevant. Among the possible attacks there are large-scale network probing activities and **DDoS** (Distributed Denial of Service). One way to defend ourselves is to detect and predict via **passive monitoring**, keeping track of the traces of attacks that are collected by the Darknets: **backscattering packets and port scans**. **Darknets** are range of advertised, but unused, IP addresses, studying the darknet traffic at our disposal we try to propose a simple way to **cluster, visualize and analyse** the spurious data. In this thesis we focus on a **complex network approach** to the problem: instead of representing the packet records in a highly dimensional euclidean space of points we create a **relationship traffic graph** on the model of a social network, formed by nodes, e.g. **IP, AS** (Autonomous System), **ports** and we isolate **communities** (strongly connected and related sub-groups) that could hide implicit information about malicious traffic. The algorithm proposed are **Label Propagation** and Greedy **Modularity** combined with a similarity measure based on the **Jaccard Similarity** between nodes inside the graphs. Once this cyber-intelligence information are inferred from the Darknet communities the future work could be to compare it with a real public-addressing scenario and use the features to isolate the malicious traffic from the huge amount of good packets exchanged.

Contents

List of Figures

Chapter 1

Introduction

1.1 Darknet Analysis

1.1.1 Internet: benefits and risks

Internet is today the most widely used technology. The environment is pervaded and the number of connected devices could soar up to 25 billions by 2020: we use it for communications, payments, healthcare, domotic, wearable etc. The benefits and risks of this technology affects not only the ICT world but has consequences on society, politics, institutions. The effect of this pervasion is the steep increase of the risks that comes from a cyber-attack executed on the network: together with the good actors there are several malicious ones that takes advantage of the vulnerabilities presents in the existing internet architecture. As the usage increases the surface of attack gets wider: one of the main way in which the attacks are performed is to abuse the weaknesses of the existing protocols used to share the data through packets. In order to protect ourselves from those attacks we can make use of one of the weapons at disposal of cyber-defenders: the analysis of the packets that cross our network.

1.1.2 Darknet: look in the shadow

Applications produce a huge amount of data that is sent in the network through packets, often in clear, conveying a small amount of information that can be used to gain control over the behaviours of the users. At this point the main problem that arises is that considering the load of the traffic that traverses the network, the useful information that offers the possibility to label a packet as malicious or prevent an attack can get lost inside the good traffic. One way to monitor the threatening traffic is to look “in the shadows” of the network were it is more likely to find packets that signal anomalies: one place of the internet where it is useful to look are **Darknets**.

The word Darknet could be misleading and it is often used in place of *Dark-Web* or *Deep-Web*, i.e. web-services that are not accessible in normal web browsing and that are hubs for illegal activities, marketplace etc. In order to solve this misunderstanding different synonym will be employed: network telescope, darkspace, blackhole monitor, network sink, Internet background radiation (IBR), unallocated but reachable IP addresses, unassigned IP addresses etc [9].

A darknet is a range of *advertised, but unused, IP addresses* [9]: the absence of any IP interaction makes it safe and simple to implement. The Darknet is a sink that collects data in order to passively monitor anomalous traffic under which a possible cyber-attack is hidden [12]: in a standard fault-free scenario it would not be possible to find these IPs as source or destination address of a sniffed packet since the hosts of the darknet have never been involved in any IP communication with other devices.

1.1.3 Cyberattacks

Why this anomalous traffic is observed? One benign reason is **misconfiguration** [2], e.g. a wrong set-up in which the IP of the darknet is inserted rather than the correct one. Another reason is a **suspect action** that lead a device into addressing a Darknet IP, producing the suspicious packet. Different kind of attacks can be the reason behind the generation of this traffic and a darknet is a powerful lens that has been used by cyber-security for detecting attacks such as **backscattering** packets generated by a victim of a DDoS [15] or a **port-scan** [13] generated by worms or botnets [3].

1.2 Research Questions

There are several tools and algorithms that can be used for IDS on traffic passively collected; moreover the data itself can be represented in different abstractions over which the algorithms can be run: unsupervised machine learning techniques, clustering on a multidimensional Euclidean space, complex network analysis of a densely connected graph etc.

One challenge was to find a simple way to represent the data that would provide a direct information on how the actors of the communication are related, without depending on a high-dimensional euclidean representation on top of which defining a complex and subjective distance definition.
Since the attacks are often driven by a central orchestrator and executed on a large scale, subjects and objects of an attack could present an implicit and strong

interconnection that could by easily inferred and used. The most intuitive way to represent the traffic flow was to build a graph of IP addresses nodes that are connected by an edge, i.e a packets of an existing connection, then we moved to a more compact and general view based on **Autonomous System** grouping of the captured traffic that would also take into account the information on the port, which itself identifies already known vulnerabilities or random probing patterns.

In order to derive useful information on traces of an attack we decided to rely on a complex network problem: **community detection**. This approach is very useful in social networks graph, where there is a graph of entities linked by different kind of relationships (a direct knowledge) and the objective is to infer the communities of entities starting only from the edges of the graph. As people belongs to different social groups, e.g. family, school, work etc., computers could belong to different communities of victims or attackers. After defining the relationships, the nodes and the graphs, another problem is to choose a proper algorithm in order to detect the clusters, testing the quality of the results and the knowledge added on the traffic. Another task is to define a similarity metric that could provide a way to simplify the construction of the relationship edges among the communities and compare the results over time.

Once the communities have been observed the task is to pinpoint a threat starting from the results of the algorithm. First of all we need to decide how to employ the information about the community id number of each node, then it is important to provide a visual meaningful representation that could directly show an attacking pattern. At this point it is important to define the time window over which the observed traffic has to be clustered and how the graph and the communities change over time, how they interact, which IP and ports are related etc.

The final important question is how this close-up insight over darknet data can be used in a real network scenario, in which the knowledge gain by the darknet has to be used as an intelligence source for the depiction of a cyber-attack.

1.3 Research Work

It is important to define a strategy once the traffic has been collected. The methodology behind the data investigation depends on:

- Data relevance
- Data representation
- Data Analysis and interpretation of results.

Different type of packets can be observed by the network telescope and based on what is the objective it is possible to exclude specific protocols. The data have been represented with a network graph, with different definition of the topology: what are the source and the sink nodes, what is the definition of an edge between the node, what are the distance metrics, etc. After filtering the data, creating the graphs, and defining the metric we choose which algorithms to use in order to analyse the graph and point out specific patterns to use as a source of intelligence on a cyber-attack.

1.3.1 Traffic Insights

First the data has been filtered, and capture where translated and formatted for correct data analysis, then were produced statistics on the type of protocols, the most active autonomous systems, the most contacted ports, the geographical position etc. A lookup of the source ip was executed in order to record the correct Autonomous system.

1.3.2 Darknet Graph Construction

At this point we decided to build three different graphs using in turn as nodes the source IPs, the destination IP, the destination ports (the targeted darknet ports), creating an edge between each nodes when a connection packet had been observe:

1. $ASN\_IP \rightarrow DST\_IP$;
2. $ASN\_IP \rightarrow DST\_Port$;
3. Port Sequence Graph

The 1^{st} and the 2^{nd} graph are both directed and unweighted, while the second is weighted.

1.3.3 Community Detection Algorithms

On top of the three graph we run two community detection algorithms:

1. **LPA (Label Propagation Algorithm)**;
2. **Greedy Modularity**

We tested the quality and the relevance of the results, producing visual representation of the traffic flow graph in order to highlights the most targeted port and the most active Autonomous Systems.

1.3.4 Community Analysis

Once the traffic was clustered we extracted the information on the membership community, produced a visual representation and look at the internal structure of each community trying to find the features typical of each group, a symptom of a possible attack. The second step is to produce a time analysis that points out how the cluster of IP change over different windows. The future is to find the same patterns of a Darknet Scenario inside the traffic collected in a public scenario.

1.4 Research Results

1.4.1 Traffic Characterization

After filtering the data we highlighted how single autonomous systems monopolize the traffic origination and specifically we highlighted the importance of certain countries or location. The traffic belonging to our Darknet **was mainly produced by Autonomous Systems and IPs located in Russia**. When we look at the protocol carried by the IP packets we found out that **more than 95% of the traffic was TCP**. Going deeper in the analysis we observed that almost all our packets carried no payload as showed by the counting of the length of each packet. The type of possible threat present in our traffic is represented by port scan or DRDoS since the traffic is mainly TCP-SYN with low number of backscattering detected.

1.4.2 Communities

We run LPA and GMA over the defined graphs and produced a graphic visualization that highlights the communities and the central role of the Autonomous Systems that generate the traffic. From this analysis we also focused on the port that are mostly scanned in order to target which kind of vulnerabilities are observed inside the community orbiting around each AS. Filtering out port that are rarely scanned for each analyzed period we could emphasize the vulnerable service and the most probed ports. Another important aim of our results was to compare how the communities and the nodes inside those communities changes over different observed period. We could find out that the centrality of certain AS system its recursive during the different time period but the output of our algorithms often produced a different composition in the internal of the clusters; the same holds for the port distribution that is always constant giving the well known vulnerabilities of certain ports but often does not provide a constant presence in the same traffic cluster over time.

1.4.3 Future Work

The future of this work is to select the meaningful results taken from our analysis and check whether this behaviours of the traffic can be found in a real network capture. The darknet is a powerful tool for focusing and tailoring the research on the features of the spurious traffic that otherwise would be lost in the large amount of the legitimate users. Is possible to find the same pattern of an attack easily inside the real traffic thanks to the community detection analysis run on the darknet? Does communities inside the Darknet occur in the communities of malicious traffic that can be found on the real network? Moreover the community detection analysis can be also seen a tool to provide pre feature to fed inside a machine learning approach that has to be build on the big data of the darknet at our disposal.

Chapter 2

Research Background

2.1 Darknet Overview

We will now provide a wider view on the concept of Darknet and on the reason behind the track observed by them. When establishing a Darknet the administrator of the network set up a group of servers with the aim of watching by stealth the traffic that reaches a specific interval of IP addresses. The chosen IPs are public but unassigned hence it could not be possible to automatically find them as destination of a packet on the public network; moreover when a Darknet is established the servers holding a specific IP cannot answer to any requests arrived. For the above reasons most of the traffic that will be collected consists in communication establishment requests, e.g. the 1st SYN packet of a TCP three way handshake. Eventually if we detect a packet with as source an IP belonging to a Darknet, it should not exist, in the same way as a packet whose destination is pointing an IP that is not assigned to any public network gateway. Often the Darknet address appears because some malicious actor masquerades its identity by spoofing another IP in order to execute its actions, someone is performing scan on random addresses or the Darknet IPs has finished in list of destination IPs. There are already several Darknets deployed, it is worth mentioning [7] and large scale darknet monitoring projects such as [**?**] CAIDA Center for Applied Internet Data Analysis.We will now try to provide an explanation on the dynamic behind the origination of large scale traffic that traverses the network and reaches the Darknet**13**.

2.1.1 Cyberattacks

Why this anomalous traffic is observed? One benign reason is misconfiguration [2], e.g. a wrong set-up in which the IP of the darknet is inserted rather than the correct one. Another reason is a suspect action that led a device into addressing a Darknet IP, producing the suspicious packet. Different kind of attacks can be the reason behind the generation of this traffc and a darknet is a powerful lens that has been used by cybersecurity for detecting attacks such as backscattering packets generated by a victim of a DDoS [15] or a port-scan **[14]** generated by worms or botnets [3].

DDoS Backscattering

DDoS (Distributed Denial of Service) is a cyber-attack that has the goal of delaying or suspending the availability of a service provided on the web. The attacker tries to create a flood of data towards the victim: the overwhelming number of packets interrupts the availability of the target. In order to hide its identity, the attacker masquerades the source IP of the packets picking random addresses. When an IP of the Darknet falls into this random choice a trace of the attack is generated Fig.2.1. For example in SYN-flooding attacks (i.e. the DoS is executed by saturating the TCP connection tables) the replies from the victim, SYN-ACK aka backscattering packets, will be collected at the Darknet IP and they will be used to infer the occurrence of a DDoS [2].

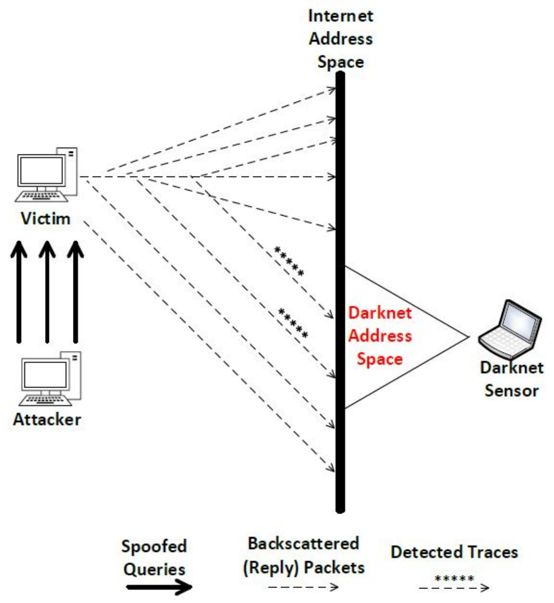


Figure 2.1: A victim of a DDoS generates replies that reach also the Darknet [9]

Port scanning

A port scan Fig.2.2 is executed when an attacker wants to gain information about the state of the ports at the receiver, understanding which of this ports are open and other important information about the vulnerabilities that can be exploited. A darknet can infer different scanning activities [8]. When an attacker, or an unconscious victim (e.g. a device infected by a worm), executes a large scale scan the probing packets can reach the network sink. It would be wise for an attacker to avoid the possibility of reaching the darknet space but this would be quite difficult considering the lack of knowledge about the existence of the silent network telescope [6].

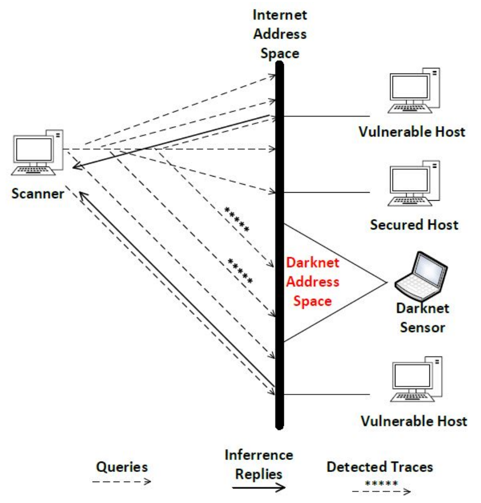


Figure 2.2: Port Scanner reaches the Darknet among the pool of scan targets

Misconfiguration

There can be several misconfiguration that create an unexpected packet reaching the network. During our work we will not focus on it but we will provide a simple intuitive scenario for sake of simplicity. One reasons behind these packets is the assignment of wrong IP addresses inside the network set-up of a network interface. Let's suppose for example that a network administrator or a technician manually insert an IP used to reach an external server that provides a specific service that is wrong or a public address is randomly inserted just for testing the functionality of a system. It of course clear that theses scenarios cannot be responsible of a consistent part of the traffic since they are much less likely to happen with respect to the previous reasons.

2.2 Community Detection

In this section we will provide a short introduction to formal concepts of graph theory that will be often used throughout the text and we will explain the importance of graph analysis on real networks. We will move forward with the explanation of the concept of community structure inside the networks and why it is important for our case of study. Finally we will introduce and explain in details the two algorithms that we choose among the ones already present in the literature: **LPA**[1] and **GMA**[2].

Graph Theory Overview

When it comes to represent a network, e.g. an highway line, the ISDN infrastructure, a neural tissues, the working hierarchy of a company, the social groups of a neighbourhood; the graph is the most logical mathematical abstraction. Formally a graph G(V; E) is defined as:

- $\boldsymbol{V}$ a set of *nodes* or *vertices*, $\boldsymbol{v} \in \boldsymbol{V}$;
- $\boldsymbol{E}$ a set of edges, $\boldsymbol{e} \in \boldsymbol{E}$, that interconnect the elements of $\boldsymbol{V}$.

The set $\boldsymbol{E}$ itself is a subset of $\boldsymbol{V^2}$, i.e. the set of node pairs inside V. This formal definition will be used in the next chapter when our case study graphs will be introduced [10].

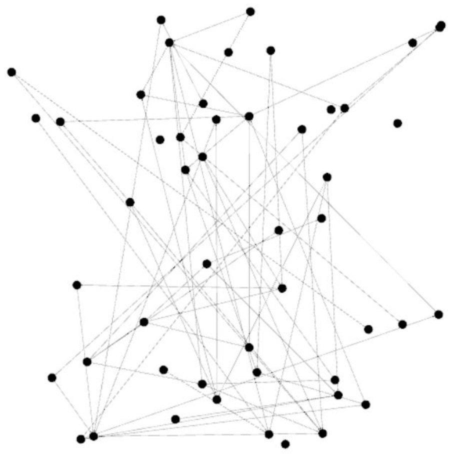


Figure 2.3: A graph with its nodes and its edges

[1]Label Propagation Algorithm

[2]Greedy Modularity Algorithm

Digraphs and Multigraphs

A graph can be **directed** or **undirected**, and in the former case we will call it a **Digraph** [10]. For example, considering an ICT scenario, a TCP connection between two nodes, e.g. two IP addresses $\boldsymbol{ip_1}$ and $\boldsymbol{ip_2}$ is directed becasue the order of the endpoints in pairs $\boldsymbol{(ip_1; ip_2)}$ and $\boldsymbol{(ip_2; ip_1)}$ represents two distinct edges $\boldsymbol{e_{ip_1,ip_2}} \in \boldsymbol{E}$ and $\boldsymbol{e_{ip_2,ip_1}} \in \boldsymbol{E}$. An example of **undirected** graph is a friendship relation between two people, in the real world or in a social media. We will provide both directed an undirected examples. When it is possible to find multiple parallel edges between two graphs, e.g two TCP connections $\boldsymbol{ip_1 \rightarrow ip_2}$ over two distinct ports, providing two different services, we are dealing with **Multigraphs** [10].

Degree and Weight

Other two important concepts that are useful to describe a graph and that will be later employed in the analysis are the **degree** of node and the **weight** of an edge. When two vertices $\boldsymbol{v_1}$ and $\boldsymbol{v_2}$ are connected by an edge $\boldsymbol{(v_1; v_2)}$ they are $\boldsymbol{neighbours}$. In general any $\boldsymbol{v_i}$ can be connected to a subset of $\boldsymbol{K_{vi}}$ vertices, $\boldsymbol{K_{vi}}$ is the degree of $\boldsymbol{v_i}$. When the graph is directed the degree can be distinguished in input or output degree, based on the direction of the adjacency relationship $\boldsymbol{v \rightarrow v_1, ..., v_n}$ or $\boldsymbol{vi \leftarrow v_1, ..., v_n}$.
When a number is associated to every existing edge, the graph is weighted otherwise is un-weighted[10].

2.2.1 Community Structure

The graph abstraction gives the possibility to perform a graph analysis in order to understand the features of the system under study [10]. In the analysis of graphs describing real networks it is important to take into account their irregularity nature because *they are objects where order coexists with disorder* [10]. When we consider random graphs the probability of existence of an edge between any two vertices is equal and this will result in an similar degree for each vertex [10]. This random behaviour is not present in a real network graph, there are certain nodes ”more important” than others. More formally we should say that the degree distribution differs and there are some nodes with high degree while others with a low degree. The previous assumption is valid on a macroscopic scale but also on a microscopic scale: edges are concentrated inside subgroups of vertices and they are sparse outside those ones. The former explanation is what we called **community structure** inside a real network[10].

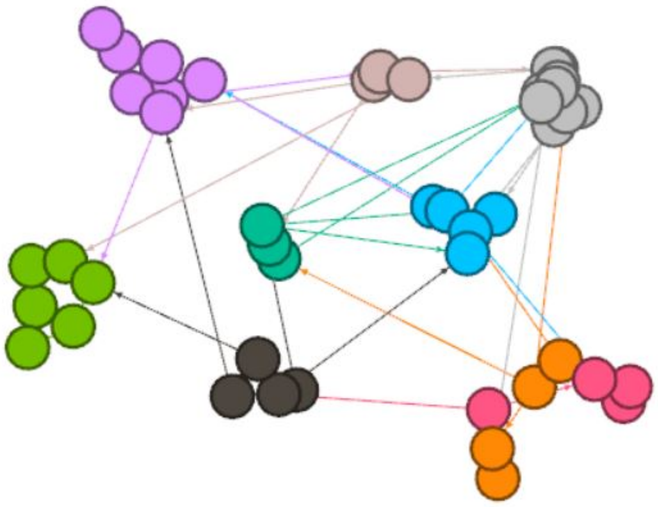


Figure 2.4: Community structures in a graph

If we look at the logical meaning of a community inside a graph we can assume that the **vertices of a community could share common characteristics** that let them belong to that specific subgroup. This is what we want to find inside our graph and it is what other researches in different fields tried to find in their specific graphs. For instance a community could be important for clustering web pages connected by reciprocal hyperlinks, social media friendship connections, protein-protein interaction, political communities inside graph of tweets etc.

Not only a community gives information about what makes the vertices part of the same subgroup but they also highlight what are the most important nodes, the ones that concentrates edges and other nodes around them. For example, we could think that a group of computers belonging to the same organization that executes a large scale scan activity would result central in the representation of a TCP connection graph. The hierarchy present inside a cluster could represent the hierarchy between victims and attacker of a cyberattack?

2.3 Community Detection Algorithms

Community detection algorithms have the goal of inferring the communities by using the information present in the graph topology [10]. After the formalization of the problem and the first attempt to define an algorithm [11] several proposals were presented. As said in the previous section we selected two choices:

1. **Label Propagation Algorithm** [17]
2. **Greedy Modularity Algorithm** [14][4]

2.3.1 Label Propagation Algorithm (LPA)

We will provide a brief explanation oof the algorithm by Raghavan et al. [?]. This paper proposed a community detection algorithm based on label propagation. LPA is based on the idea that a node $\boldsymbol{v}$ that has k neighbours $\boldsymbol{v_1, v_2, ..., v_k}$, each with a label denoting its community, decides to which community it belongs based on the label with majority among the ones of its neighbours. More formally in a network with:

- $\boldsymbol{G(V;E)}$ graph;
- $\boldsymbol{|V| = n}$ nodes;
- $\boldsymbol{|E| = m}$ edges;

the aim of any community detection algorithm is to find the subgroups of nodes denoted with $\boldsymbol{C_i}$, e.g $\boldsymbol{n}$ distinct $\boldsymbol{C - 1, C - 2, ..., C_n}$.

In the initialization phase, every vertex $\boldsymbol{v \in V}$ has a unique label that will propagate through the network. As they propagate, densely connected groups of nodes quickly reach a consensus on a unique label. The dense groups of nodes that share the same label continue to expand outwards until it is possible. At the end of the propagation process, nodes having the same labels are grouped together as one community [17].

In general at the $\boldsymbol{n^{th}}$ iteration a vertex $\boldsymbol{v_x}$ updates its label based on the labels of its neighbours at iteration **n - 1**:

$$\boldsymbol{C_{v_x}(n) = f(C_{v_{i1}}(n), ..., C_{v_{im}}(n), C_{v_{i(m+1)}, ..., C_{v_{ik}}(n-1)})} \tag{2.1}$$

$\boldsymbol{v_{i1}, .., v_{im}}$ are neighbours of $\boldsymbol{v_x}$ that have already been updated at $\boldsymbol{n}$ while $\boldsymbol{C_{v_{i(m+1)}}, ..., C_{v_{ik}}}$ are the community labels assigned at iteration $\boldsymbol{n - 1}$ to the neighbours that are not updated yet at time $\boldsymbol{n}$. $\boldsymbol{C_{v_x}(n)}$ is the label of node $\boldsymbol{v_x}$ at time $\boldsymbol{n}$. The LPA steps are:

1. Initialize every node $\boldsymbol{v} \in \boldsymbol{V}$ with a unique label $\boldsymbol{l_v}$: $\boldsymbol{C_v(0) = l_v}$;
2. $\boldsymbol{n = 1}$;
3. Arrange the nodes in the network in a random order and set it to $\boldsymbol{X}$;
4. For each $\boldsymbol{v_x in V}$ chosen in that specific order $\boldsymbol{X}$, execute function 2.1 that will return the label occurring with the highest frequency among neighbours; ties are broken uniformly randomly.
5. Repeat from step 3 until all vertices has the majority label of its neighbors.

The reason behind the use of labels is that with their propagation through the network densely connected groups of nodes form a consensus on their labels. At the end of the algorithm, nodes having the same labels are grouped together as communities [17]. LPA is recommended for its simplicity and time efficiency: it is not based on any quality function optimization on communities but it just use the information already present in the network structure. It does not require to indicate at the beginning the number of desired communities or their size, this information will be the output of the algorithm. The time complexity of each iteration of the algorithm is $\boldsymbol{O(m)}$, the number of iterations to convergence appears independent of the graph size, or growing very slowly with it [**10**]. This makes the algorithm useful for the analysis of large systems like the one we are dealing with in our scenario. The algorithm can produce overlapping communities. This is not necessarily a drawback but would introduce the need of a solution in case of ties. Furthermore on the drawback of LPA is the tendency of the algorithm to return a single community when dealing with strongly interconnected graphs with large large number of edges.

2.3.2 Greedy Modularity Algorithm (GMA)

The second algorithm that we will explain is a greedy version of the modularity optimization algorithm. First the concept of modularity will be explained and then based on *Blondel et. al* [4] we will provide the formalization of the problem.

Modularity

Modularity is first defined as a metric for identifying the quality of partitions resulting from clustering. Described by *Girvan and Newman* [11]. The idea is that a random graph is not expected to have a cluster structure in which edges and nodes are condensed, so comparing the real graph with a random graph of equal size it possible to infer a cluster. The algorithm looks at the density of edges in a sub-graph and at expected density of the sub-graph belonging to same graph but with vertices reattached randomly [10].

In order to calculate the expected density a *null model*, i.e. a random graph, has to be defined. The null model is a copy of the original in the sense that it keeps its structural properties but breaks its community structures.
Modularity can be written in general as:

$$\boldsymbol{Q = \frac{1}{2m}\sum_{ij}(A_{ij} - P_{ij}) \cdot \delta(C_i, C_j)} \tag{2.2}$$

where:

- $\boldsymbol{i, j \in V}$ are any pair of vertices;
- $\boldsymbol{A_{ij}}$ the element at $\boldsymbol{i^{th}}$ row and $\boldsymbol{j^{th}}$ column of the **adjacency matrix A**;
- $\boldsymbol{P_{ij}}$ the expected number of edges between vertices i and j in the null model;
- $\boldsymbol{\delta = 1}$ if $\boldsymbol{C_i = C_j}$ (same community); $\boldsymbol{\delta} = 0$ otherwise.

The generation of the random graph can be autonomously and arbitrarily defined but it has to respect the degree distribution of the original graph. The standard null model of modularity imposes that the expected degree sequence (after averaging over all possible configurations of the model) matches the actual degree sequence of the graph [10]. The null model definition follows the one provided by [1].

Let's consider a generic node $\boldsymbol{v_x \in V}$ and two other nodes $\boldsymbol{v_i}$ and $\boldsymbol{v_j}$, with respectively degrees $\boldsymbol{K_i}$ and $\boldsymbol{K_j}$ to which $\boldsymbol{v_x}$ could be connected. In order to respect the degree of each node, the edges are not removed but "cut" in half [10]. In order to form an edge between $\boldsymbol{v_i}$ and $\boldsymbol{v_j}$, half-edges incident with $\boldsymbol{v_i}$ and $\boldsymbol{v_j}$

have to be joined. We do not provide the proof for the expression of the parameter in 2.2 $P_{ij} = 2m \cdot p_i p_j$, resulting from the edge rearranging process.

Considering that δ removes all the contributions of nodes outside the same community the 2.2 can be re-written as:

$$Q = \sum_{c=1}^{M} \left[\left(\frac{l_c}{L} \right) - \left(\frac{d_c}{2m} \right) \right] \tag{2.3}$$

where the sum is over the M communities in the graph, l_c is the number of links inside community c, d_c is the total degree of the nodes in the community C. In the equation we can see the comparison between the real fraction of edges in the cluster $\frac{l_c}{L}$ and the expected fraction of edges $\frac{d_c}{2m}$ of a random graph that respects the degree distribution of the starting scenario.

Greedy Modularity Optimization

The concept behind the Modularity optimization is that a better division of the graph into clusters corresponds to high modularity values . Detecting communities than becomes an optimization problem over a quality function Q that in our case corresponds to the modularity. Since finding the maximum value of modularity among all the possible partition of a graph is computational NP-complete we will consider a greedy method provided by Newman [16] and its application [5] implemented in the python library Networkx used in this work. The algorithm can be summarized in the following steps:

1. In the initial phase any v_x among $|V| = n$ vertexes is the unique member of a community. These means that there are n communities.

2. Join communities in pairs and calculate the increase of Q;

3. Merge the pairs that provide the greatest increase or the lowest decrease in Q;

4. Repeat 2 and 3 until the modularity Q reaches the maximal value, i.e. there is no possible merge that increase Q;

The algorithm is useful because it can scale well to large networks with a computational time that is usually $O(m+n)n$ or $O(n^2)$ in the worst case. Moreover it can be extended to weighted graphs. Considering the size of our dataset and the possibility to provide a weighted and un-weighted version of each graph we found the GMA implementation by NetworkX and Gephi feasible for our work.

2.3.3 Jaccard Similiarity

We will briefly explain the Jaccard Score, a metric that has been employed in our analysis and it is commonly used as a statistic for measure similarity and diversity of sets. The reason behind this choice is that it is also possible to see a community as a group of nodes that share some similarities hence it is needed to choose and define a metric. In order to introduce the Jaccard Score we will make also a useful example that anticipates the use that we could make of the metric in our study, let's consider a graph G(V; E) and two nodes:

- $\boldsymbol{v_i}$ and $\boldsymbol{v_j}$ that represents two source addresses $\boldsymbol{ip_i}$ and $\boldsymbol{ip_j}$;
- $\boldsymbol{N_i}$ and $\boldsymbol{N_j}$, respectively the neighbours set of $\boldsymbol{ip_i}$ and $\boldsymbol{ip_j}$;
- $\boldsymbol{N_i = p_{i1}, .., , p_{ik}}$ and $\boldsymbol{N_j = p_{j1}, .., , p_{jm}}$ represent the destination ports of a TCP or UDP connection;
- $\boldsymbol{ip_1 \rightarrow p_{i1}}$ means $\boldsymbol{ip_1}$ contacts the port $\boldsymbol{p_{i1}}$.

We are interested in calculate how similar are ip1 and ip2 by using the Jaccard Score as:

$$\boldsymbol{J(v_i, v_j) = \frac{|N_i \cap N_j|}{|N_i \cup N_j|}} \tag{2.4}$$

The Jaccard score can be defined in different ways. For example it could offer the possibility to compare how communities over time change and how they overlap based on the defined metric. A cluster consisting in a central autonomous systems and a group of ports that present an high similarity ratio with a different cluster detected in a future time window could be seen for example as an attack pattern towards infected victims that are unaware of the hidden and repetitive activity that is reaching their device. In conclusion the Jaccard Score is useful for detect and evaluate groups of nodes.

Chapter 3

Methodology

3.1 Datasets

In order to analyse the traffic we needed to collect the data that comes from an in-campus Darknet that was already deployed for research purpose.
We had at our disposal a database of captures covering a time window of three months from January 2019 to March 2019. The sniffed packets were collected in .pcap files of 1 hour traffic inside the darknet, i.e. the smallest time window is 1 hour.
At this point of the research we had to choose the approach and the amount of data to analyse in order to produce the Community graphs over which running our analysis. A single 1h capture weights $\sim$ 25 MB, meaning that for a complete analysis of a month we should deal with a load of 18 GB. At this point we could not take into account the idea of performing an analysis over the entire three month database since the study was not performed distributively but on a local machine. We decided to take samples of the data, e.g. considering specific days for each month and sampled hours during the day taking into account a reasonable time span that would try to avoid correlation between the activities performed in consecutive captures. We obtained for a single month a $\sim$ 600 MB data size.

An example of this sampling process for general day in January is:

```
01/2019
   ├── 01/01/2019
   │      ├── 1:00 A.M
   │      ├── 10:00 A.M
   │      ├── 5:00 P.M
   │      └── 11:00 P.M
   ├── 08/01/2019
   │      └── ...
   ├── 14/01/2019
   │      └── ...
   └── 28/01/2019
          └── ...
```

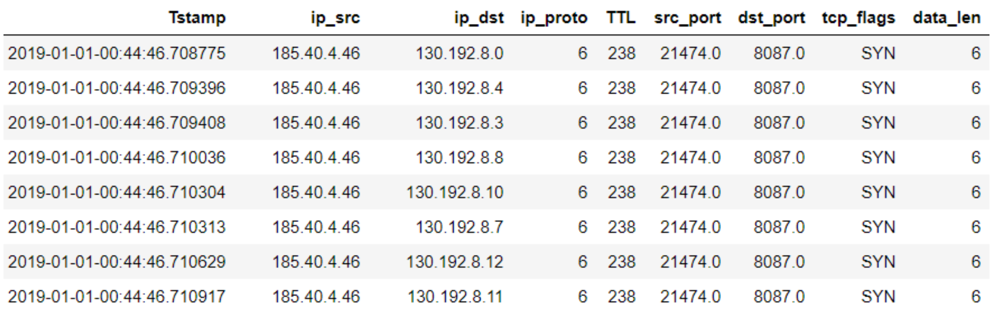
We will see in the following section how this trade-off still arise a problem of scalability and noise in the dataset, that was faced with a different approach based on the knowledge retrieved from the characterization of the data. Furthermore this choice has also the aim of diversifying the research from the already existing studies that take into account large amount of data, packet inspection with semantic and syntactic analysis of the payload, and computationally costing learning algorithm. In a nutshell we privileged passive monitoring, graph based, empirical analysis to active monitoring, high complex inspection, since our aim is to obtain behavioural cyber-intelligence focused on "who" and not on "what".

After choosing the files of our interest we had the problem of extracting from the packet capture the information stored in the header and provide a file format that would be human readable, feasible for data analysis tools for inspection and characterization of the traffic.

We filtered the .pcap file using Pcapy[1] a Python extension module that enables software written in Python to access the routines from the pcap packet capture library. The script had also the role of filtering out from the traffic the packets that were not of our interest. The final collected protocols are:

- IPv4 packets;
- TCP, UDP, ICMP protocol carried by IPv4.

After the first step the data is represented as table of packet entries 3.1, a row for each header with the following attributes:



Tstamp	ip_src	ip_dst	ip_proto	TTL	src_port	dst_port	tcp_flags	data_len
2019-01-01-00:44:46.708775	185.40.4.46	130.192.8.0	6	238	21474.0	8087.0	SYN	6
2019-01-01-00:44:46.709396	185.40.4.46	130.192.8.4	6	238	21474.0	8087.0	SYN	6
2019-01-01-00:44:46.709408	185.40.4.46	130.192.8.3	6	238	21474.0	8087.0	SYN	6
2019-01-01-00:44:46.710036	185.40.4.46	130.192.8.8	6	238	21474.0	8087.0	SYN	6
2019-01-01-00:44:46.710304	185.40.4.46	130.192.8.10	6	238	21474.0	8087.0	SYN	6
2019-01-01-00:44:46.710313	185.40.4.46	130.192.8.7	6	238	21474.0	8087.0	SYN	6
2019-01-01-00:44:46.710629	185.40.4.46	130.192.8.12	6	238	21474.0	8087.0	SYN	6
2019-01-01-00:44:46.710917	185.40.4.46	130.192.8.11	6	238	21474.0	8087.0	SYN	6

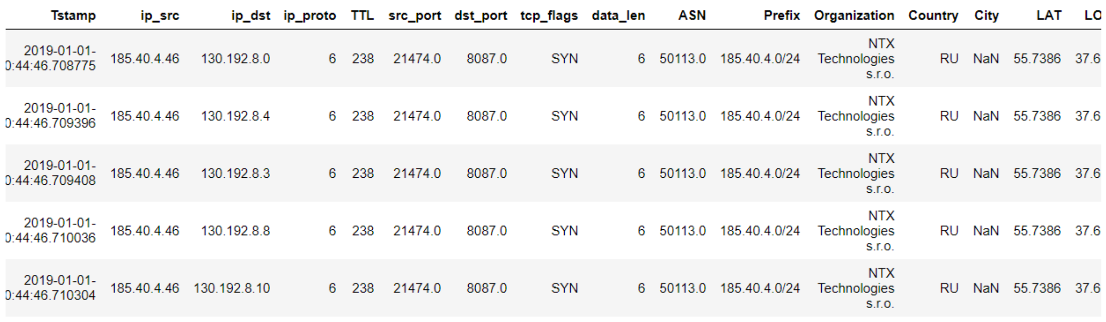
Figure 3.1: Initial data as Pandas Dataframe

[1]https://github.com/helpsystems/pcapy

At this point we lacked geographical information, useful in order to characterize the Darknet traffic and aggregate the packets based on common features derived for each $\boldsymbol{ip_{src}}$. In order to do so we made use of two specific python modules Maxmind GeoLite2[2] that provide free to use downloadable databases for IP lookup and and pyasn[3] a Python extension module that enables very fast IP address to Autonomous System Number lookups. Without the possibility of grouping entries by ASN it would not be possible to build the graph and perform a meaningful and scalable analysis of it, as we will show in the following sections. GeoLite2 gave us important geographical information on th IP source. Summarizing:

- Pyasn: ASN, Organization Name, IP Prefix;
- GeoLite2: City, Country, Latitude and Longitude.

After wrapping all the attributes in a final .csv file we started our data analysis using Pandas on Jupyter Notebook. The starting enriched dataframe has the following structure:



Tstamp	ip_src	ip_dst	ip_proto	TTL	src_port	dst_port	tcp_flags	data_len	ASN	Prefix	Organization	Country	City	LAT	LO
2019-01-01- 0:44:46.708775	185.40.4.46	130.192.8.0	6	238	21474.0	8087.0	SYN	6	50113.0	185.40.4.0/24	NTX Technologies s.r.o.	RU	NaN	55.7386	37.6
2019-01-01- 0:44:46.709396	185.40.4.46	130.192.8.4	6	238	21474.0	8087.0	SYN	6	50113.0	185.40.4.0/24	NTX Technologies s.r.o.	RU	NaN	55.7386	37.6
2019-01-01- 0:44:46.709408	185.40.4.46	130.192.8.3	6	238	21474.0	8087.0	SYN	6	50113.0	185.40.4.0/24	NTX Technologies s.r.o.	RU	NaN	55.7386	37.6
2019-01-01- 0:44:46.710036	185.40.4.46	130.192.8.8	6	238	21474.0	8087.0	SYN	6	50113.0	185.40.4.0/24	NTX Technologies s.r.o.	RU	NaN	55.7386	37.6
2019-01-01- 0:44:46.710304	185.40.4.46	130.192.8.10	6	238	21474.0	8087.0	SYN	6	50113.0	185.40.4.0/24	NTX Technologies s.r.o.	RU	NaN	55.7386	37.6

Figure 3.2: Dataframe

[2]https://www.maxmind.com
[3]https://github.com/hadiasghari/pyasn

3.1.1 Darknet Characterization

After the first phase of our research work we started to use the information at our disposal in order to profile the traffic collected and retrieve useful information as a guideline for the next steps. We produced numerical statistics that pointed out most frequent features among the attributes present in the dataframe, e.g. frequent destination ports, most active ASN, typical packets, most used protocols, payload length etc. and we used the geographical fields to build a map of the location that produced more traffic. All the results will be presented in the last chapter.

Here we will provide an explanation on the reason behind this characterization. First of all we needed a general view on the traffic in order to have a starting point from which we could compare the consistency of the results of the algorithm towards the raw statistic of the database, that were empirical and not precessed but just extracted. The second reason behind this characterization is to find attribute that could be helpful to aggregate data in order to reduce the overhead of the packet that feeds the graphs.

Based on the layer of the TCP/IP stack we could provide different definition of source and destination. As we will see this very logical concept would be at the base of the definition of the vertexes in our graphs.

3.2 Capture Graphs

The crucial part of our research was to define a graph abstraction for the provided dataset. We started from the most intuitive representation of the internet traffic, i.e an $\boldsymbol{IP} \longleftrightarrow \boldsymbol{IP}$ communication, and then moved to more compact versions in order to reduce the size of the graphs. In general we provided three version that can be tuned and changed for the sake of visualization or for reducing computational time of the algorithm:

- External IP $\rightarrow$ Darknet IP;
- AS Number $\rightarrow$ Destination Port
- Port sequence graph

The next sections will provide the formal definition of each graph. In the next chapters will be presented the results of LPA or GMA executed them.

3.2.1 IP $\rightarrow$ IP

As mentioned before this is the most logical representation of Internet traffic and it is our first attempt of graph analysis. The following graph in Fi.g4.3 will be mentioned as $\boldsymbol{ip} \rightarrow \boldsymbol{ip}$. Let's consider a digraph $\boldsymbol{G(V;E)}$:

- $\boldsymbol{G}$ is **un-weighted**;
- $\boldsymbol{G}$ is **unidirectional**;
- $\boldsymbol{v_i} \in \boldsymbol{V}$ is a generic vertex representing an IP address recorded in the packet header;
- $\boldsymbol{e} \in \boldsymbol{E}$ is an edge that connects $\boldsymbol{ip_{AS}} \rightarrow \boldsymbol{ip_{DN}}$ an external IP source to an IP destination internal to the Darknet.

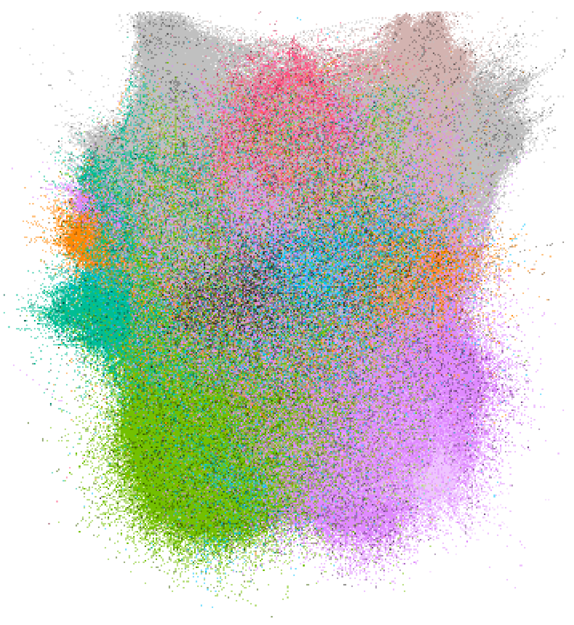


Figure 3.3: The noisy $\boldsymbol{ip} \longrightarrow \boldsymbol{ip}$ makes the computation harder and the visualization difficult

G has only one direction because the IP internal to the darknet are set up in order to not respond to any incoming request. They work in stealth mode and no one in the public network can receive any traffic from them, that is why they are also name sink. G is un-weighted but it is possible to assign to each edge $\boldsymbol{e} \in \boldsymbol{E}$ an attribute or label that indicates on which destination port the connection was targeted; moreover the source nodes could be labelled with an attribute that indicates the ASN to which the IP belongs. The construction of $\boldsymbol{ip} \longrightarrow \boldsymbol{ip}$ especially on wide time capture has an overwhelming size that is infeasible but it was a logical starting point. We then moved from this graph by considering an aggregated version that took into account the source ASN and the destination port.

3.2.2 ASN $\rightarrow$ Port

Considering that the goal of our analysis is to underline large scale attacks we assumed that most of the traffic is produced by a large number of devices (a wide interval of IPs) belonging to the same Autonomous System. Furthermore our Darknet is not an active actor in the communication, cannot suffer any damage from the traffic, hence it is not useful to pinpoint which specific IP is contacted but it is much more important to focus on what is the destination port of the communication.

Gaining knowledge on the port gives information on the vulnerabilities that are targeted or to the most frequent scanning pattern.
Taking into account the above assumptions we moved from $\boldsymbol{ip \rightarrow ip}$ to $\boldsymbol{AS \rightarrow Port}$ graph shown in Fig.3.4.
Here we provide the formal definition of $\boldsymbol{G(V;E)}$:

- G is **un-weighted**;
- G is **unidirectional**;
- $\boldsymbol{v_i} \in \boldsymbol{V}$ is a generic vertex representing:
  - Autonomous System Number (ASN), a source node;
  - TCP/UDP Destination Port in the L4 packet header, a destination node;
- $\boldsymbol{e} \in \boldsymbol{E}$ is an edge that connects $\boldsymbol{AS} \rightarrow \boldsymbol{p_{DN}}$ an external AS source to a Port internal to the Darknet.

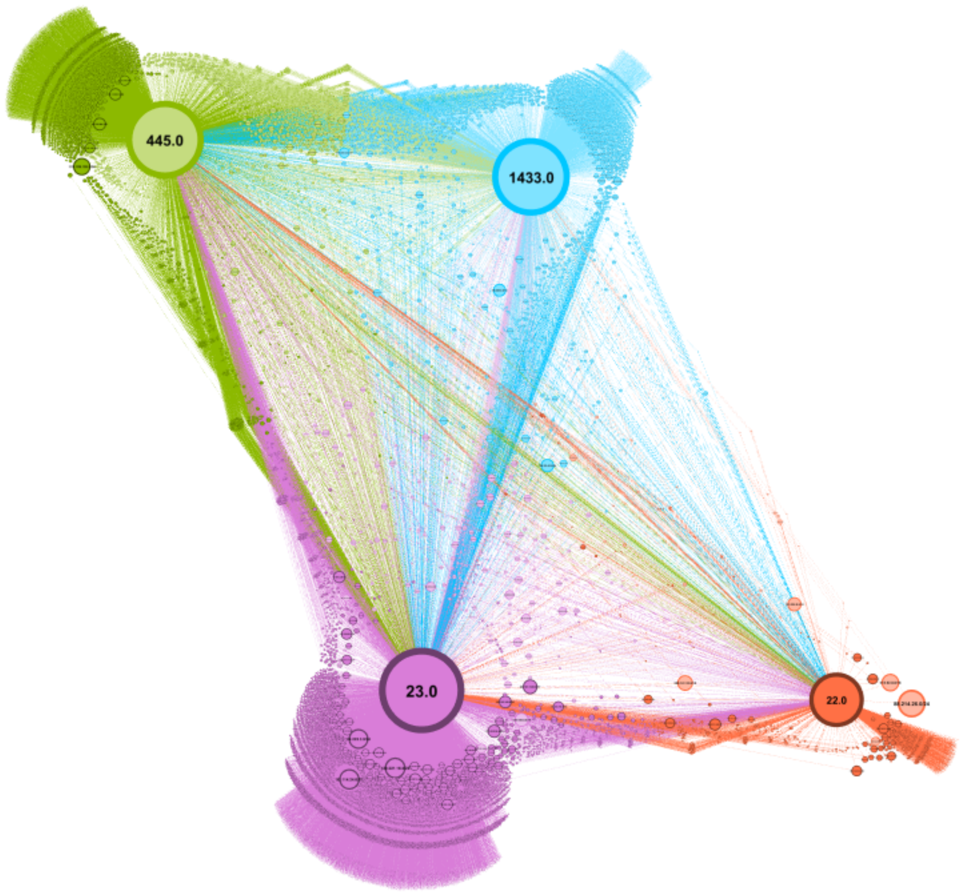


Figure 3.4: Generic $\boldsymbol{AS} \rightarrow \boldsymbol{Port}$ example

$\boldsymbol{AS} \rightarrow \boldsymbol{P}$ has only one direction for the same reason explained in the previous section. Again G is un-weighted but it is possible to assign to each edge $\boldsymbol{e} \in \boldsymbol{E}$ several attributes at our disposal from dataset. Our main focus was on $\boldsymbol{AS} \rightarrow \boldsymbol{P}$ over which we could run faster the chosen community detection algorithms and assign more relevance to the observation of the ports inside our traffic.

Before proceeding it is important to provide a clarification on the results obtained in the communities. When detecting a community we group vertexes that can be differentiated between Autonomous Systems and ports. If we consider a generic packet entry $\boldsymbol{PKT(ASN,\ IP_{src},\ P_{src},\ IP_{dst}, P_{dst})}$ we will notice that based on the value present in $\boldsymbol{ASN}$ and $\boldsymbol{P_{dst}}$ the packet community $\boldsymbol{C_{PKT}}$:

$$\boldsymbol{C_{PKT}} = \begin{cases} \boldsymbol{C_i} & \text{if } \boldsymbol{ASN_{PKT}} \in \boldsymbol{C_i} \\ \boldsymbol{C_j} & \text{if } \boldsymbol{P_{dst_{PKT}}} \in \boldsymbol{C_j} \end{cases} \quad \boldsymbol{C_i = C_j} \ \vee \ \boldsymbol{C_i \neq C_j}. \tag{3.1}$$

When $\boldsymbol{C_i} \not\equiv \boldsymbol{C_j}$ a problem arises since a packet would have a collision in the choice of the community to which it belongs. This would also add a drawback for the traffic analysis starting from the obtained communities. In order to overcome this problem we have added two more attributes the dataset. Instead of considering a single attribute $\boldsymbol{C_{id}}$ we will considered, $\boldsymbol{Port\ C_{id}}$ and $\boldsymbol{ASN\ C_{id}}$. As we will see in the Results, this difference will not affect the analysis since the collision between $\boldsymbol{Port\ C_{id}}$ and $\boldsymbol{ASN\ C_{id}}$ will be numerically irrelevant.

3.2.3 Port Scan Graph

Port Sequences

Let P be the set of all TCP ports. Assuming, a source and destination IP address, a sequence is:

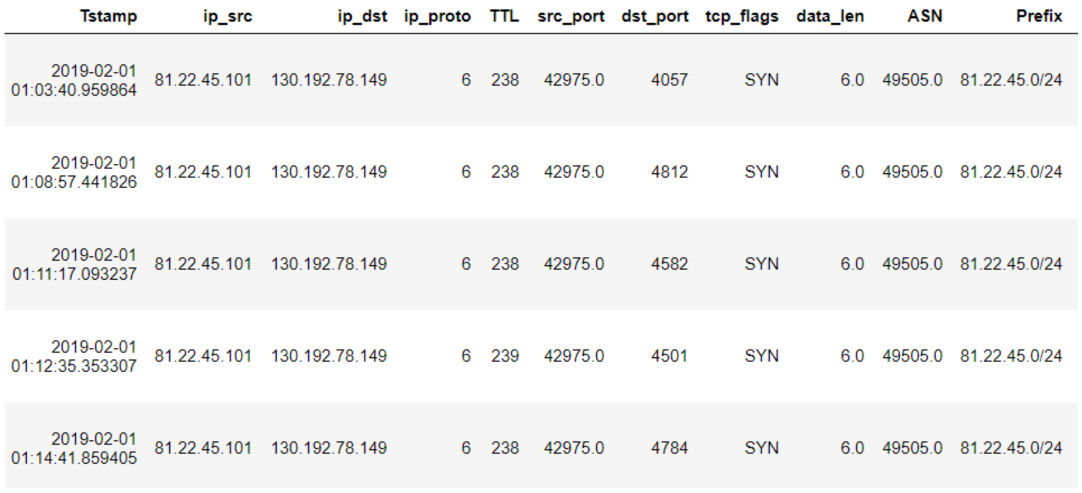
$$\boldsymbol{S((IP_{src}, IP_{dst}), T_s, T_e)} \tag{3.2}$$

the sequence of targeted ports by $\boldsymbol{IPsrc}$ to $\boldsymbol{IPdst}$ between the starting time Ts and the ending time Te. Port-scan graph over a set of type of ports V is a graph $\boldsymbol{G = (V, E, \beta)}$:

- $\boldsymbol{V}$ is the set of observed port numbers from from $\boldsymbol{P}$.
- $\boldsymbol{E}$ is a set of edges in $\boldsymbol{G}$. Let $\boldsymbol{p_u}$ and $\boldsymbol{p_v}$ be be two port numbers in $\boldsymbol{V}$;
- There is an edge $\boldsymbol{(p_u, p_v)}$ if there is a port scan sequence in which the two ports are adjacent, i.e. scanned in sequence;
- $\boldsymbol{\beta}$ is a function that assigns for each edge $\boldsymbol{(p_u, p_v)}$ the number of dependency occurrence;

The graph is built in order to take into account the information that is truncated from the previous analysis and that here becomes crucial in order to derive frequent port-scan pattern. While $\boldsymbol{AS \rightarrow Port}$ is focused on who performs the scans here we focus in what are port-scan. In the previous definition we analysed how the importance of port 23, (which represent 8% of the packets roughly 160.000 packets on a dataset of 2 millions entry) is different from ports 50000 and 50001 (which together represent less than 0.002%). But when deal with port-scan patterns taking apart destination 50000 and 50001 could result wrong.

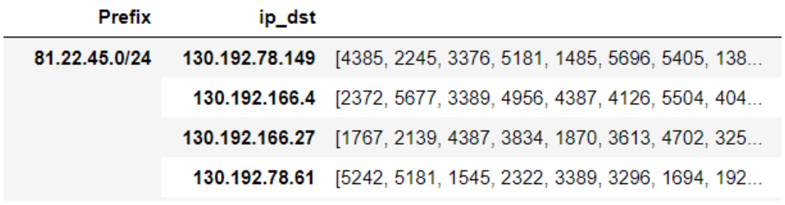
We will provide an example of the building process of the graphs. Let's look at a generic port sequences on 1st of February 2019 between T_s = 1:00 A.M. and T_e = 1:59:59 A.M. from source IP 81.22.45.101 to destination IP 130.192.78.149 Fig.3.5:



Tstamp	ip_src	ip_dst	ip_proto	TTL	src_port	dst_port	tcp_flags	data_len	ASN	Prefix
2019-02-01 01:03:40.959864	81.22.45.101	130.192.78.149	6	238	42975.0	4057	SYN	6.0	49505.0	81.22.45.0/24
2019-02-01 01:08:57.441826	81.22.45.101	130.192.78.149	6	238	42975.0	4812	SYN	6.0	49505.0	81.22.45.0/24
2019-02-01 01:11:17.093237	81.22.45.101	130.192.78.149	6	238	42975.0	4582	SYN	6.0	49505.0	81.22.45.0/24
2019-02-01 01:12:35.353307	81.22.45.101	130.192.78.149	6	239	42975.0	4501	SYN	6.0	49505.0	81.22.45.0/24
2019-02-01 01:14:41.859405	81.22.45.101	130.192.78.149	6	238	42975.0	4784	SYN	6.0	49505.0	81.22.45.0/24

Figure 3.5: Port Scan Initial Dataframe

In Fig.3.5 ports 4057, 4812, 45802, 4501, 4784 would be filtered out since their occurrence is considered too low. In this case they will not focus on the ports by itself but they will be grouped in scan sequences whose relevance will be taken as a whole group (port sequences in Fig3.6). In this list the importance of port 23 would be the same as port 30000, so it is not feasible to delete un-frequent ports.



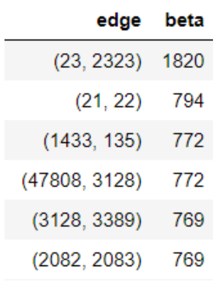
Prefix	ip_dst	
81.22.45.0/24	**130.192.78.149**	[4385, 2245, 3376, 5181, 1485, 5696, 5405, 138...
	130.192.166.4	[2372, 5677, 3389, 4956, 4387, 4126, 5504, 404...
	130.192.166.27	[1767, 2139, 4387, 3834, 1870, 3613, 4702, 325...
	130.192.78.61	[5242, 5181, 1545, 2322, 3389, 3296, 1694, 192...

Figure 3.6: Port Sequence Snapshot

The result is a list of port sequences for each Autonomous System. At this point we still need to define an edge table. In order to do this we delete from our dataset the port sequences that consist in only few ports so we set as threshold a length of 5 elements.

Most of this sequences consists in single scan performed with high frequency over relevant destinations such as 80,23,445 etc. This loss is not important for us since it is completely taken into account in $ASN \rightarrow Port$; moreover it lightens the graph dimensions.

At this point we create the edge table as explained in the formalization of the problem at the beginning of this section obtaining for example an edge table like in Fig. 3.7.



edge	beta
(23, 2323)	1820
(21, 22)	794
(1433, 135)	772
(47808, 3128)	772
(3128, 3389)	769
(2082, 2083)	769

Figure 3.7: Port Scan Graph Edge Table

Chapter 4

Results

4.1 Darknet Traffic Charcterization

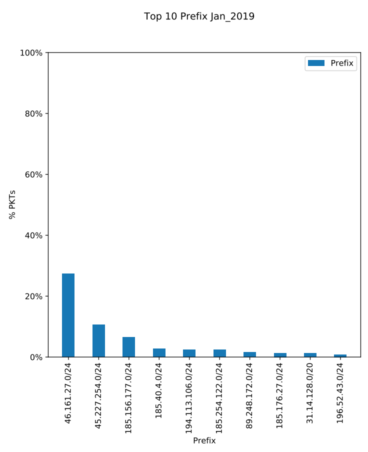
In this section we will provide the results of statistics collected for each time window of analysed captures. We will present respectively:

- Top 10 ASN;
- Top 10 Organizations;
- Geographical Distribution;
- Top 10 Destination Ports;
- IP Protocols and TCP Flags;

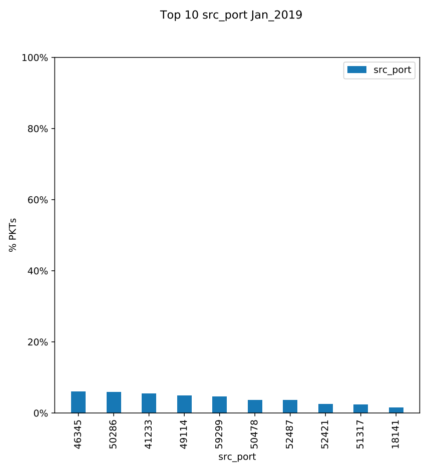
The results will be compared between January. February and March 2019, and a small summary will be presented at the end.

4.1.1 Darknet Filtering

The following results will point out the consequences of the filtering process for the study. For example let's look at the following 4 metrics in % of packets for each source ASN Fig.4.1a, source port Fig.4.1b; destination port Fig.4.2a and IP destination Fig.4.2b, in January 2019:

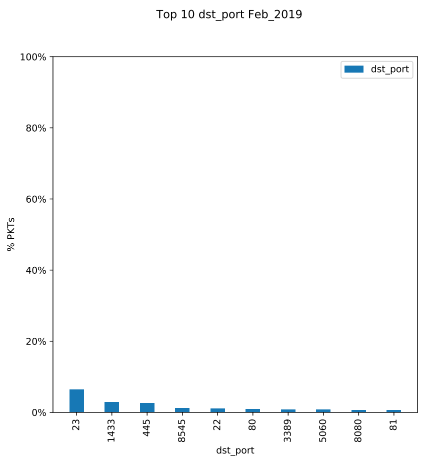


(a) Source Prefix

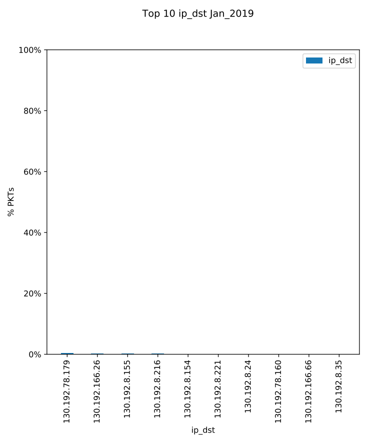


(b) Source Port

Figure 4.1: Prefix/Port Comparison



(a) Destination Port



(b) IP Destination

Figure 4.2: Port/IP Comparison

From the Fig.4.1 we can see how sparse is the distribution of packets by source ports compared to the ASN Prefix; moreover we can make the same conclusion for destination ports and IP destination Fig.4.2. Defining a graph like $P_{src} \rightarrow P_{dst}$ or $AS \rightarrow IP_{dst}$ based on what we have seen in this characterization phase would result in a high number of vertexes and edges that would affect negatively the community detection analysis.

If we still look at Fig.Fig.4.2a we would notice that the top 10 destination ports hardly reach 20% of the whole traffic. This means that there is a long tail of ports that appears in less than 0.1% of packets, i.e. with an amount of information that is not useful for our analysis but that consist in 80% of the dataset when considered as a whole. The effect of this noisy packets is to cover and slow down the analysis that should concentrate its force on those packets that carry out meaningful and consistent information. We have decided to filter out those ports since it is very important on the effect of the community detection. Even after the filtering we want to underline that in the analysis we were still considering ports that appeared in $\sim$ **0.001%**, i.e. $\sim$ 200 over a total **2 million** entries.

The visual effect of this filter can be provided by comparing a community detection graph unfiltered like in Fig.4.3 and Fig.4.4:

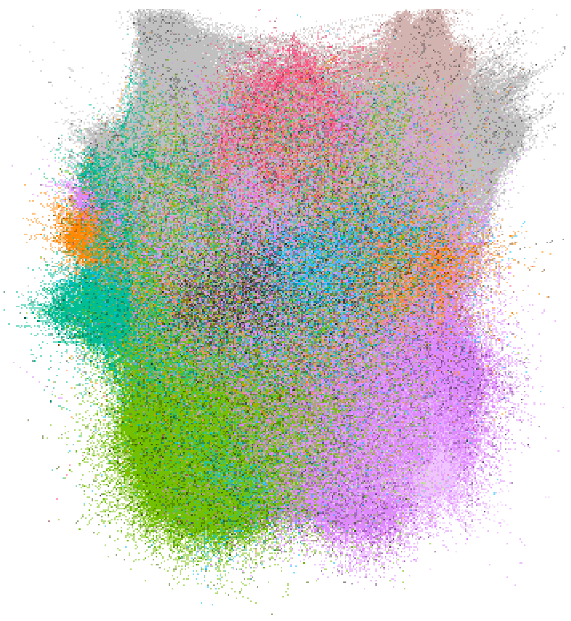


Figure 4.3: Unfiltered Graph

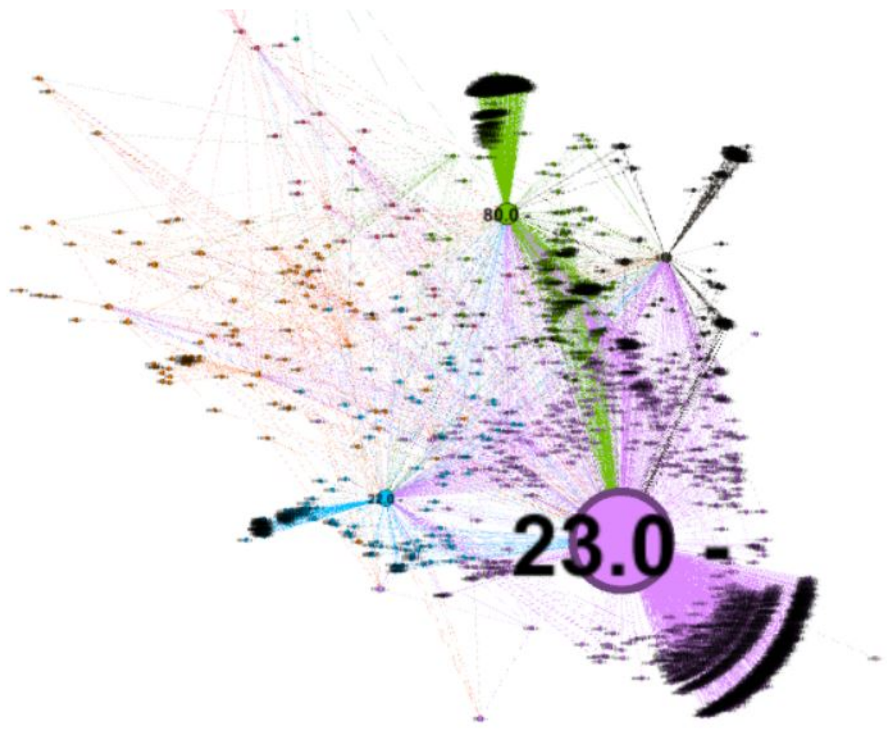
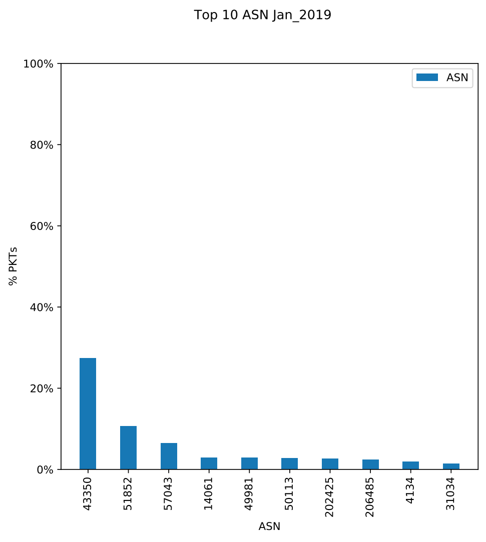


Figure 4.4: Filtered $ip \rightarrow ip$

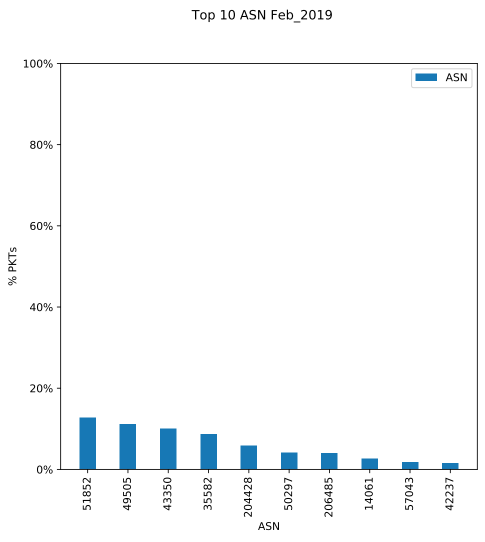
The noisy consisting in a large amount of edges and vertexes inside the graph makes even from qualitative point of view (visually) unfeasible. On the other hand this approach cuts out a numerous part of the traffic. We decided to do not lose this knowledge and reinsert in the definition of the **Port Scan Graph**.

4.1.2 Top 10 ASN

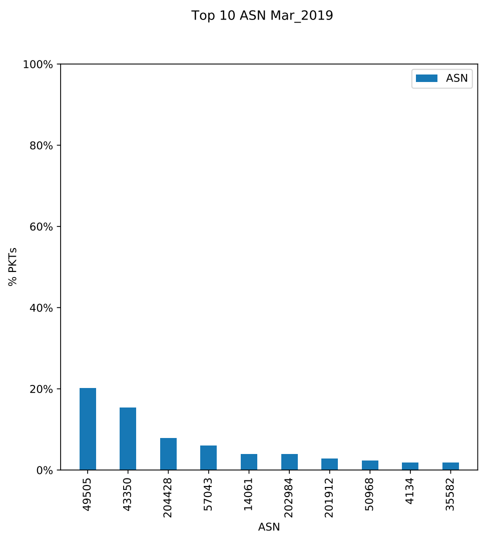
The following Fig.4.5a,4.5b,4.5c shows the % of packets in the dataset containing a specific Autonomous System Number in the ASN attribute field obtained in the pre-processing look-up phase with pyasn for each observed month:



(a) Top 10 ASN Jan 2019



(b) Top 10 ASN Feb 2019

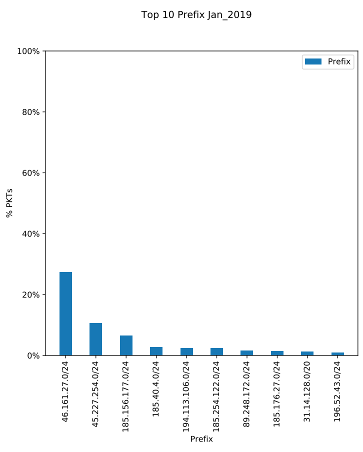


(c) Top 10 ASN Mar 2019

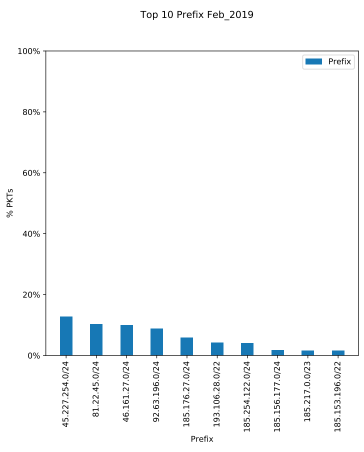
Figure 4.5: ASN Comparison

First of all we have to point out an important fact, all the 3 captures consists of approximately 2 million rows. Taking this in mind we can see for example that

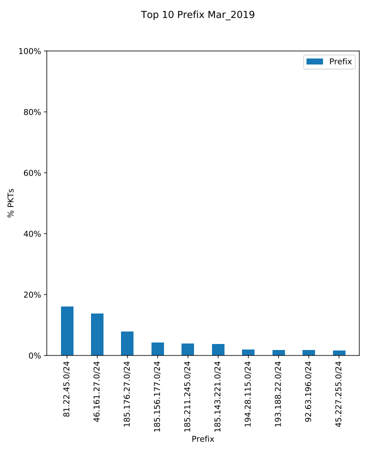
in Jan 2019 4.5a ASN 43350 is responsible for the origin of 27% of the packets, that means approximately 550K entries. Taking into account the top 3 ASN we can see that they cover 40% of the total traffic. Three actors are responsible of consistent part of the communication, a similar behaviour can be seen for February and March too Fig.4.5b, Fig.4.5c. These results suggest us two conclusion, one is that the majority of the spurious traffic is caused by few actors that operate on large scale, the second is that those AS candidate themselves as central nodes in the community detection analysis. Furthermore we can see that AS 43350, top 1 for Jan, is present also for Feb top 2, and March top 2. Same consideration can be done for other top ASN. This suggest to us that we should expect for some of the community that we will find a grade of similarity between the three observed time windows.
Since the lookup depends on the accuracy of the third party database we will provide here the top 10 Prefix, in order to show that is still possible to perform an analysis aggregating flow by Prefix without loss of compactness. This trade-off is evidenced by the collected results and the difference is not relevant or affect the consistency of the results.



(a) Top 10 Prefix Jan 2019



(b) Top 10 Prefix Feb 2019



(c) Top 10 Prefix Mar 2019

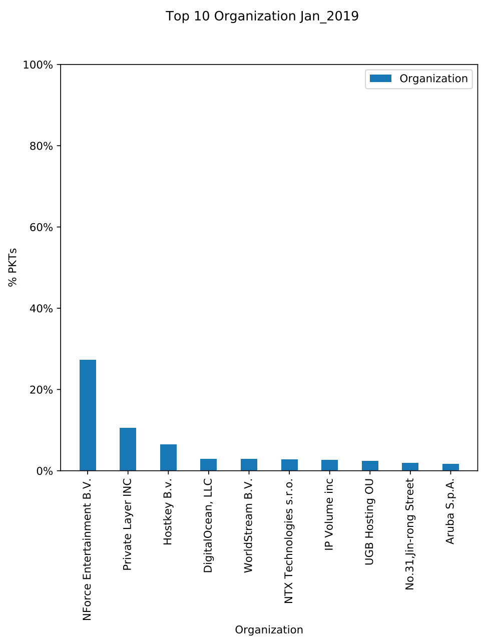
Figure 4.6: Prefix Comparison

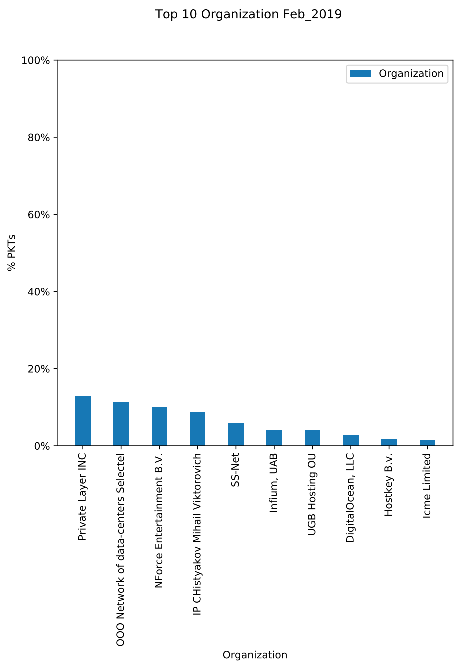
4.1.3 Top 10 Organizations

In the next page we present in Fig.4.7 the top 10 Organizations in terms of % of packets for the three observed time windows. We found that:

- NForce Entertainment B.V. is the most active Organization with 26%, followed by Private Layer INC and Hostkey B.V. for January 2019;
- Private Layer INC, OOO Network of data-centers Selectel, NForce Entertainment B.V. for February 2019 with %15 each;
- OOO Network of data-centers Selectel %20 for March 2019 and NForce Entertainment B.V. with 18%.

As we said some organization appears more frequently in the captures, in general showing a distribution in packet similar to the one provided for the ASN and the Prefixes. We should again recall that this organization information rely on the precision of a third party database that we trusted but were not taken into account as input for the community detection algorithm. We acknowledged that few organizations, few AS owning Prefixes, produces majority of the spurious traffic giving confidence on the idea that the community detection analysis should isolate the spurious AS that produce this junk traffic. At this point we should also profile the geographical distribution of the dirty packets.





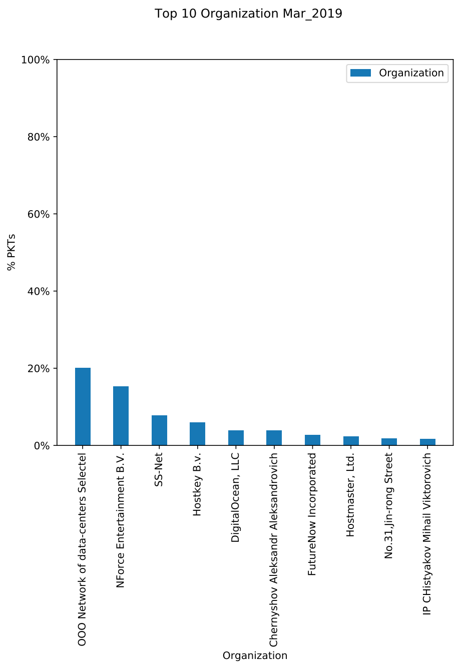


Figure 4.7: Org. Comparison: (a) Jan, (b) Feb, (c) Mar

4.1.4 Geographical Distribution

The following graphs will provide a numerical and a visual representation of the locations which produce more traffic. The geographical data are indicative and subjective to the third-party databases. They are relevant for the statics and profiling of the traffic but will not affect in any way the algorithm since they were not fed to them. First we show in Fig.4.8, Fig.4.9, Fig.4.10 the scatter plot of % of packets produced for every tuple of latitude and longitude coordinates.

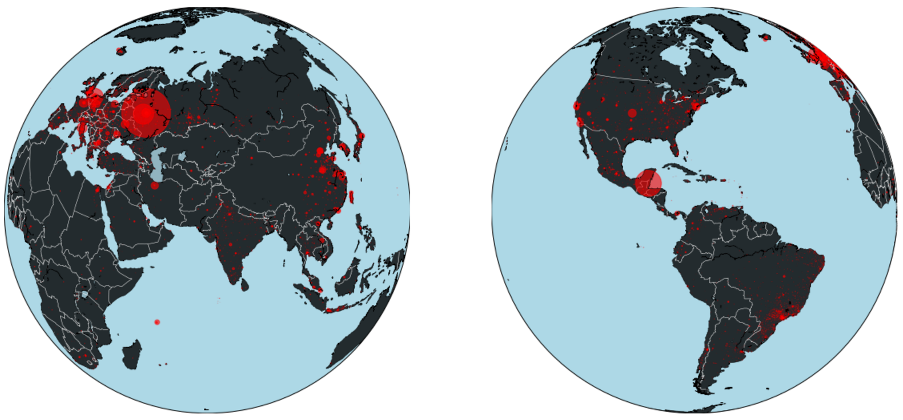


Figure 4.8: Jan 2019 Scatter

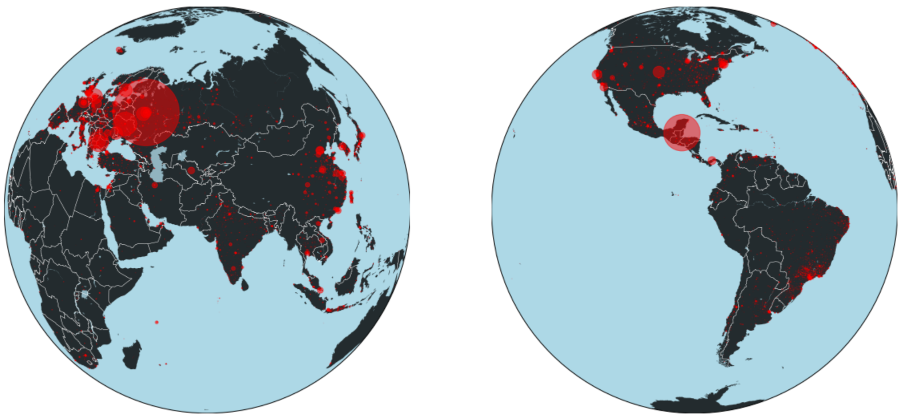


Figure 4.9: Feb 2019 Scatter

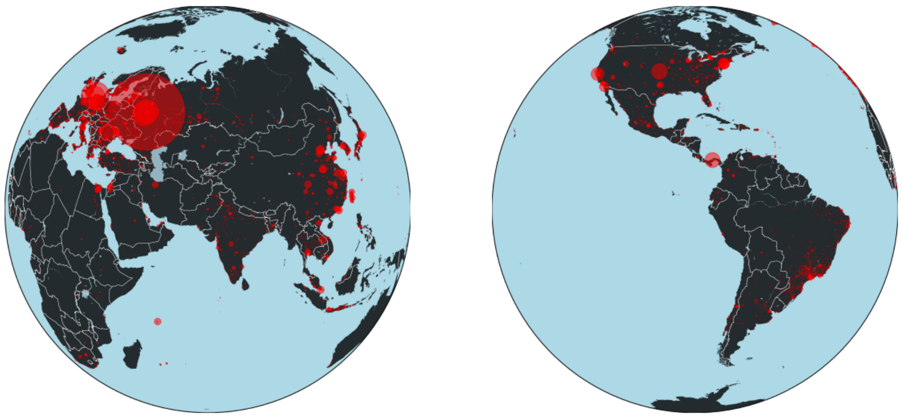
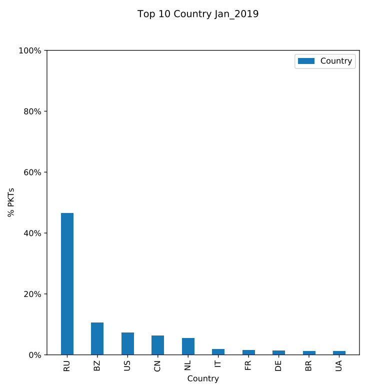


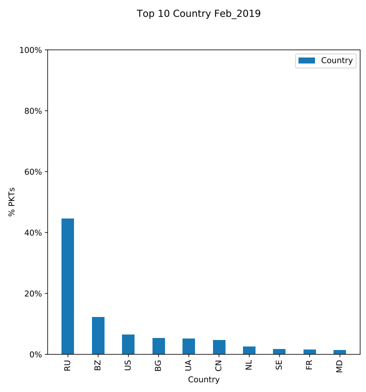
Figure 4.10: Mar 2019 Scatter

First we can see how the geographical source location of the packets does not change in time. Another conclusion is that it clear that a central role in the production of the data is played by Europe and specifically by Russia and Netherlands. Other hotspots of the traffic can be found in Mexico, China, Brazil and main capitol cities of the US. This result would satisfy the initial expectation and the common sense that specific location would be responsible for the production of spurious traffic. Furthermore taking into account the statistic collected on the IPs Prefixes, AS, and organization this reinforce the starting assumption of profiling the malicious actors using the darknet. At this point the information are still globally observed, in the Community Detection section we will go more underneath the surface of this characterization.

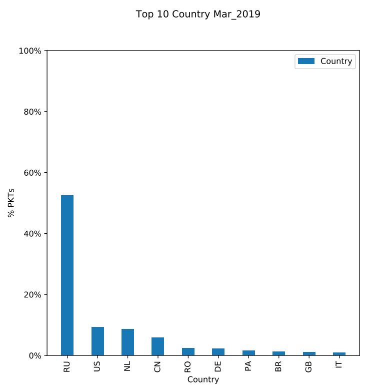
In order to get a numerically representation we can look at Fig.4.11



(a)



(b)

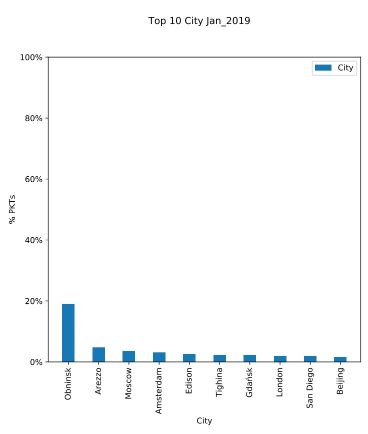


(c)

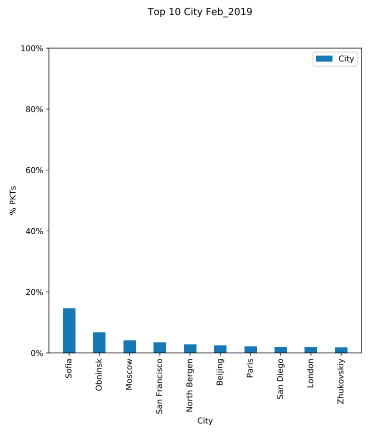
Figure 4.11: Country Comparison

Numbers collected for the Country attribute field confirm what is provided by the coordinate scatter plots. Russia is responsible for almost 50% of the spurious traffic that we sniffed in the Darknet. This assumption is constant during time, and same can be said for Brazil (BZ), US, China (CN) and The Netherlands (NL).

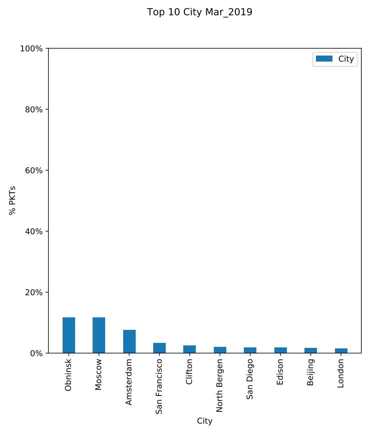
Just for an illustrative purpose we will also show the most observed cities in the City attribute fields. This metrics is not considered relevant since the lookup process for the cities may be suffer of imprecision and often resulted in a null result. Anyway it can still be interesting to look at this Fig.4.12:



(a)



(b)



(c)

Figure 4.12: City Comparison

We see that Obinsk is often an active hub of traffic, the city located in the western part of Russia. This result strengthen the previous results that showed an important scatter value in the correspondent geographical area. A role in this correspondence is also played by Sofia and Moscow. Another point in favour of a better AS profiling.

Geographical Distribution Summary

The geographical analysis highlights the following results:

- A consistent part of the traffic is centralized by specific groups of IP belonging to specific AS;
- The same behaviours is observed when looking at the Organizations owning the IPs;
- The scatter plot together with the % bar plot and the cities traffic distribution makes it clear that the sources of the spurious traffic are not various, and they have a clear hotspot, at least for our observation period and from our darknet, in Russia or in general in Eastern Europe, especially from Obinsk, Moscow and Sofia.

4.1.5 Top 10 Destination Ports

We present the top 10 Destination Ports in terms of % of packets for the three observed time windows:

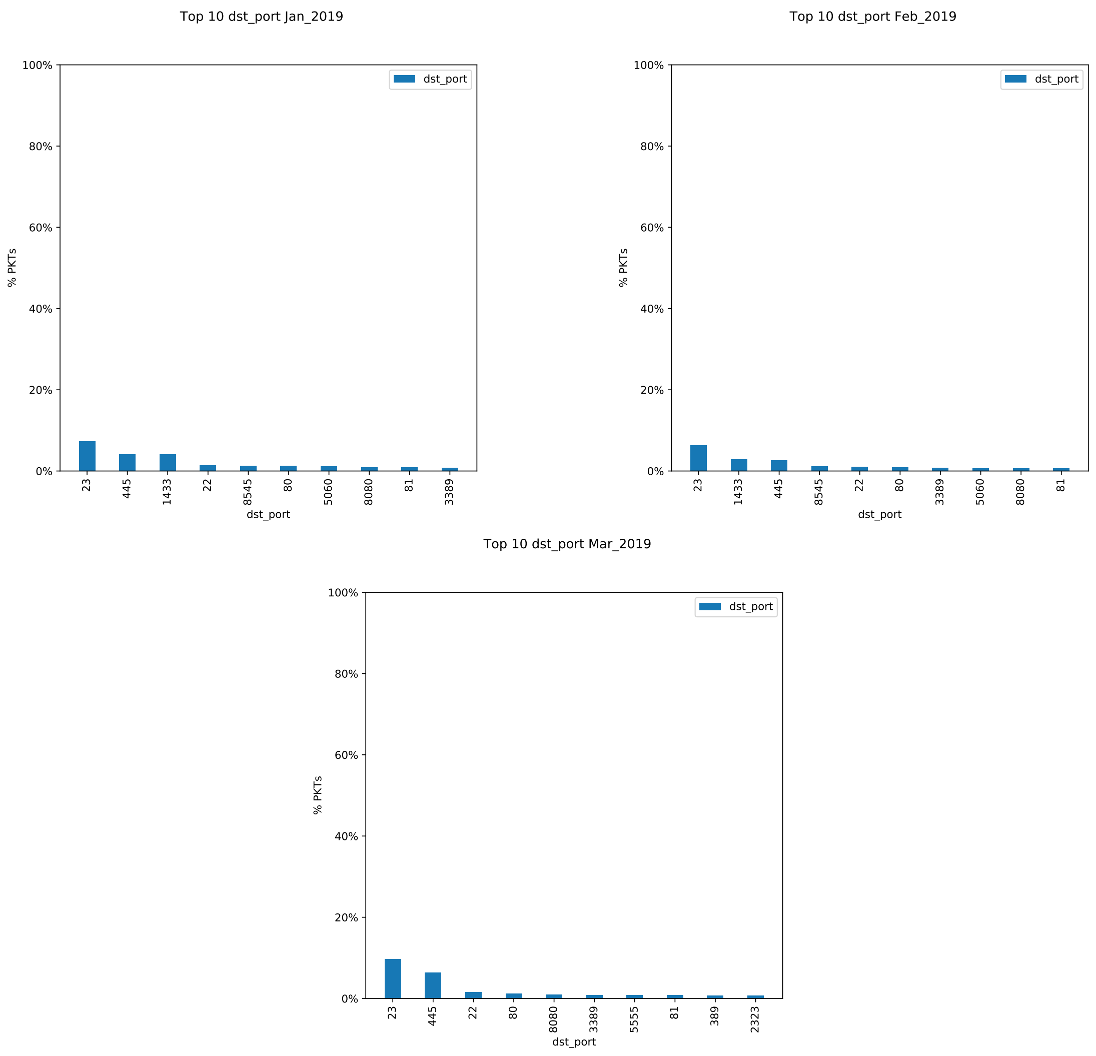


Figure 4.13: Destination Port Comparison

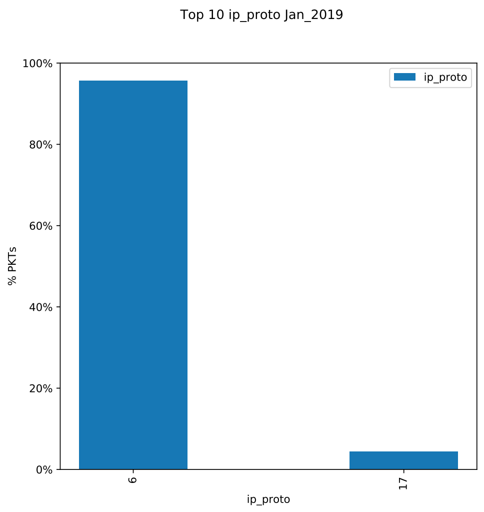
As we can see from Fig.4.13 we found that the most frequent destination port in the packets are the one we expected since in general to this numbers are linked widely used applications that are popular for their serious vulnerabilities found over the years:

- 23 Telnet;
- 445 microsoft-ds;
- 22 SSH;
- 80, 81, 8080 HTTP;
- 1433 ms-sql-s (Microsoft SQL Server).

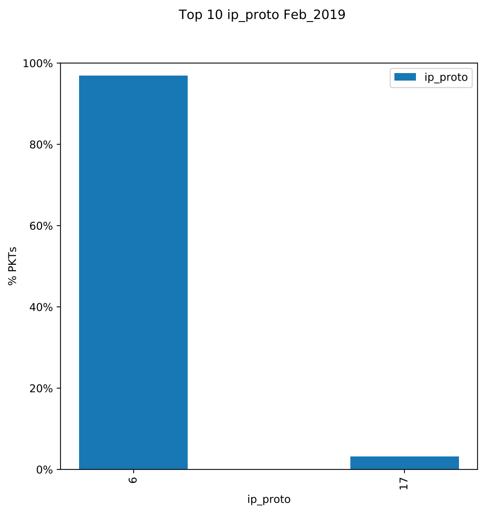
These port are common throughout the the time. They were used as filter towards the big dataset from which we have built the $ASN \rightarrow Port$ graph. More precisely we deleted from the dataset those ports appearing for less than 0.001% of the packets, i.e. 200 hits over a dataset of 2 million rows. This allowed us to save almost 80% of the noisy packet entries.

4.1.6 IP Protocols and TCP Flags

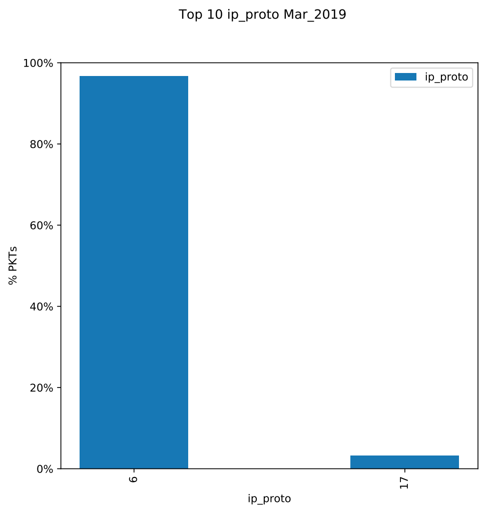
The following two figures Fig.4.14 and Fig.4.15 show the % of packets divided by IP protocol type and TCP flags:



(a)



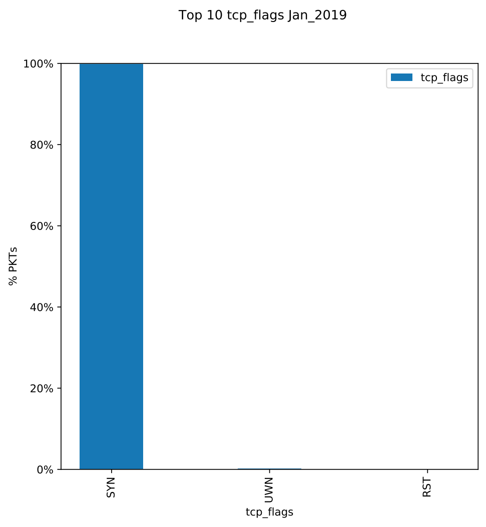
(b)



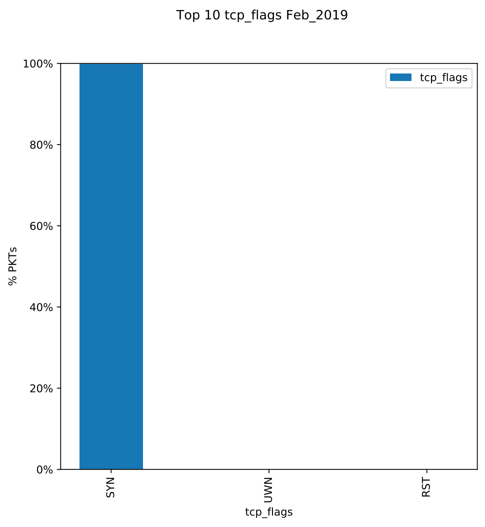
(c)

Figure 4.14: IP Protocol Comparison

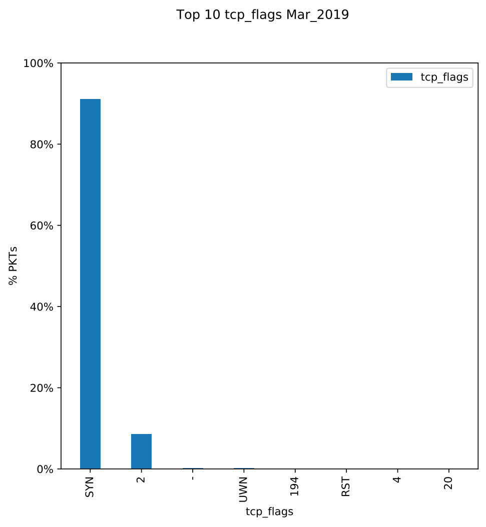
As we can see 99% of the traffic consist in TCP SYN packets. i.e. the standard packet format used to establish a TCP communication, so probing the availability of a port.



(a)



(b)



(c)

Figure 4.15: TCP Flags Comparison

Darknet Characterization Summary

In a nutshell we have seen that our dataset have a constant behaviour that can be categorized as follows:

- 99% of packets are TCP SYN probings;
- Traffic is centralized on the hand of 3 or 4 AS systems for each time windows, with some of them appearing more than once throughout the tree observations;
- Packets are often originated from the same location by active countries in Europe (mostly Russia and Netherlands), US, Brazil, China etc.
- Common port that hide vulnerabilities are the most contacted, for example:.
  - 23 Telnet;
  - 445 microsoft-ds;
  - 22 SSH;
  - 80, 81, 8080 HTTP;
  - 1433 ms-sql-s (Microsoft SQL Server).

This information will be our guideline for the results obtained by the Community Detection analysis.

4.2 ASN → Port

In this section we will provide the result of our Community Detection analysis on the $\boldsymbol{AS \rightarrow Port}$ graph. At first we will provide statics on the global results obtained for each time period, than we will provide the visual representation of the communites explaining the meaning of specific topologies visible in the network then we will dive into the communities highlighting suspicious traffic behaviours as a final result. The second part of these section will be focused on showing the time similarities between the three graphs built through the use of the Jaccard Similarity measure. We will explain the results and eventually look for specific common communities that triggered our attention.
Before proceeding we recall to take in mind the distinction between $\boldsymbol{Port\ C_{id}}$ and $\boldsymbol{ASN\ C_{id}}$ made in 3.1 in the Methodology chapter. The understanding of this distinction is important in order to properly observe the results. In order to distinguish the $\boldsymbol{C_i}$ detcted for the three intervals we will refer to them as:

$$\boldsymbol{C_{month}} = \begin{cases} \boldsymbol{A_i} & \text{if} \quad \text{January} \\ \boldsymbol{B_i} & \text{if} \quad \text{February} \quad \text{with } \boldsymbol{i = 1, 2, 3, ...} \\ \boldsymbol{C_i} & \text{if} \quad \text{March} \end{cases} \tag{4.1}$$

4.2.1 LPA Unfeasibility

At one point of our research we found ourselves in front of the choice of excluding from the analysis the use of the Label Propagation Algorithm since under several circumstances and trials, tuning the filtering parameters it resulted often in a trivial central community with irrelevant side ones. This behaviour it's clear in the following Fig.4.16:

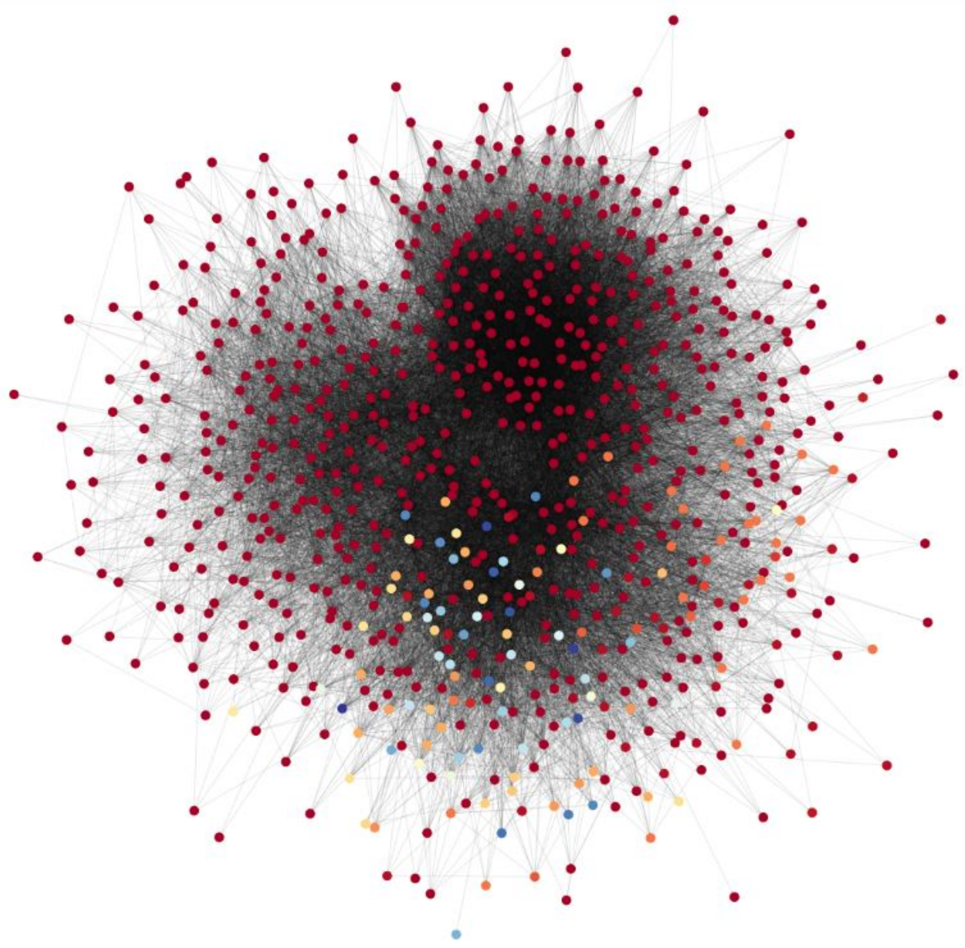


Figure 4.16: Trivial LPA single community

From now on we will refer to the Community Detection analysis only considering the GMA version provided by NetworkX and Gephi.

4.2.2 Communities January 2019

We run the GMA[1] community detection algorithm over the sampled dataset representing the month of January 2019. The following graphs show the % of packets belonging to each community distinguished by the two labels $\boldsymbol{Port\ C_{id}}$ and $\boldsymbol{ASN\ C_{id}}$.

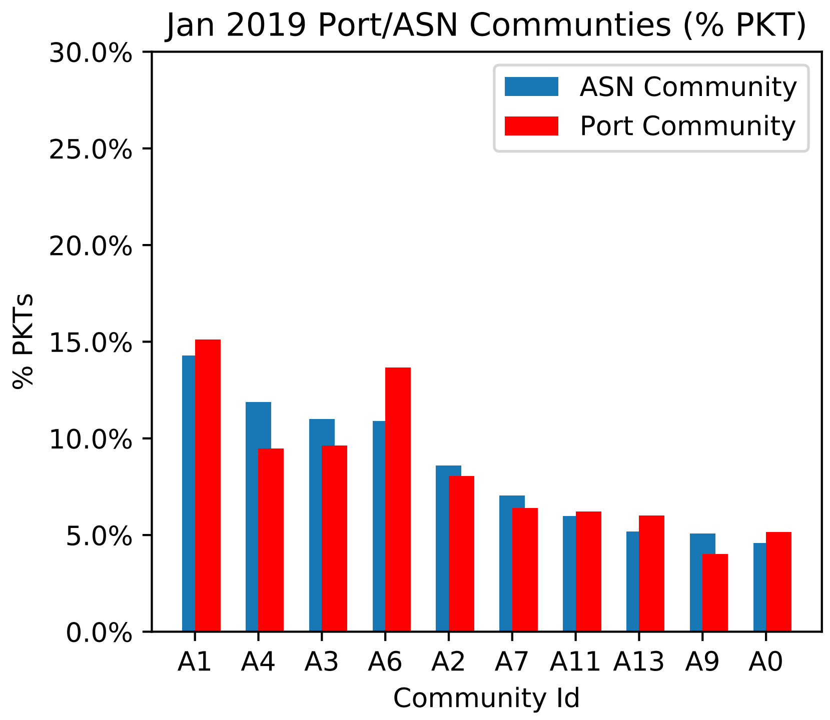


Figure 4.17: ASN/Port Communities Jan 2019

We can see from Fig.4.17 that the % of packets that present a conflict in the community identification label $\boldsymbol{C_i \not\equiv C_j}$ is not relevant. The top 10 communities wraps $\sim$ **80%** of the filtered dataset. As we will se from the next section we can already highlight four the top four communities grouping 50% of the traffic ($\boldsymbol{A_1, A_4, A_3, A_6}$).

[1]Greedy Modularity Algorithm

Graph January 2019

The visual representation of the Community graph obtained with Gephi for January 2019 is Fig.4.18:

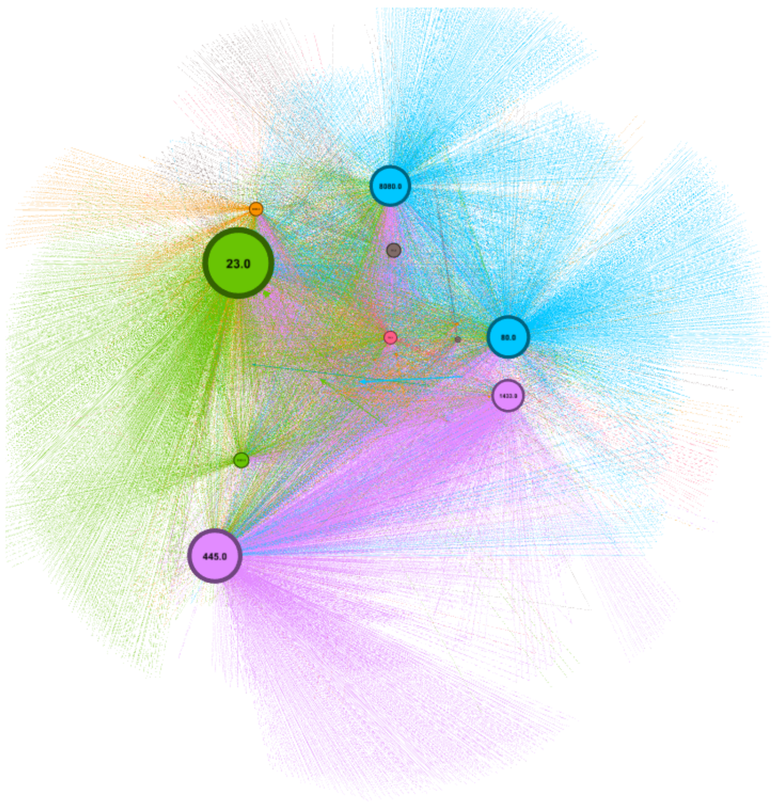


Figure 4.18: Community Graph Jan 2019

As we can se from the graph, scaled by the input-degree of the nodes, i.e. number of incoming edges, it is possible to distinguish clearly three communities, each of them consisting of source vertexes, the AS, orbiting around the centre of attraction, i.e. the port. The most frequent ports, appears bigger and share a great amount of edges: this means that the same ASN is targeting both ports. This should not be seen as an unexpected behaviour moreover since the algorithm cannot allow overlapping communities. Edges are coloured by the colour of the vertex from which they are originated, so we could say that we can distinguish the community probing 80 and 8080, one wrapping 23, another one wrapping 1433 and 445 ports over which usually run microsft services.

Community Analysis Jan 2019

After plotting the graph and the % of packets of the dataset in the community we decide to look inside each of them and extract numerical and behavioural conclusions.

Before commenting the graphs we will explain how the statistics were collected. We recall that each packet is represented with a row that contains two community labels $\boldsymbol{Port\ C_{id}}$ and $\boldsymbol{ASN\ C_{id}}$.
Thank to this expedient we can take apart the AS and the port attribute for a packet entry inside each community and overcome problems of collision (see Eq.3.1), and see how the two attribute interacts: all the following graphs will report the % of packets inside each community for each $\boldsymbol{IP} \in \boldsymbol{C_{ASN}}$ and each $\boldsymbol{P} \in \boldsymbol{C_{Port}}$, i.e. they are not % over the total dataset but they are internal distributions among labelled packets.

First of all we will point out some specific traffic pattern that triggered out attention. Let's recall the meaning of vertical and horizontal scan: the former is a scan performed over a range of ports on a single IP address, the latter is a probing on a specific port over a range of IP.

If we look at Fig4.19 we can see that IP **185.40.4.0/24** represent the % 99 of IP labelled inside $\boldsymbol{ASN\ A_0}$ while the ports inside $\boldsymbol{Port\ A_0}$ have a flat distribution of 1% for each value. We could say that this behaviour has a vertical scan pattern seen by the point og view of the performer, since in general this definition hold for the subject of the scan:

```
185.40.4.0/24
    ├── Port:  50000
    ├── Port:  60000
    ├── Port:  8087
    ├── ...
    ├── Port:  8488
    ├── ...
    ├── Port:  50004
    └── ...
```

Same conclusions can be made for $\boldsymbol{A_0, A_2, A_9, A_{21}}$:

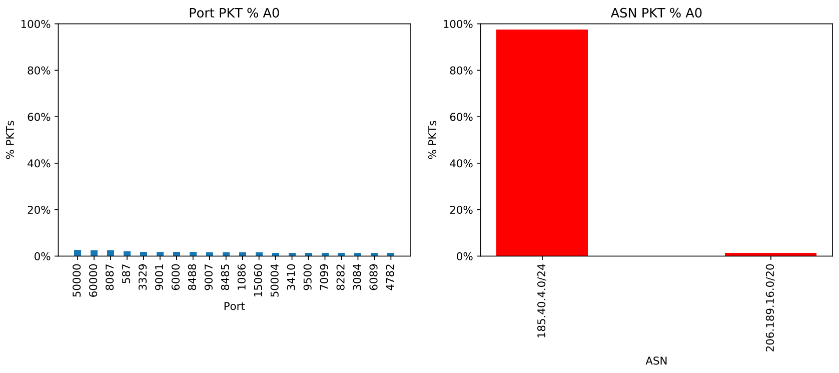


Figure 4.19: $\boldsymbol{A_0}$ consists of a single AS scanning over high ports

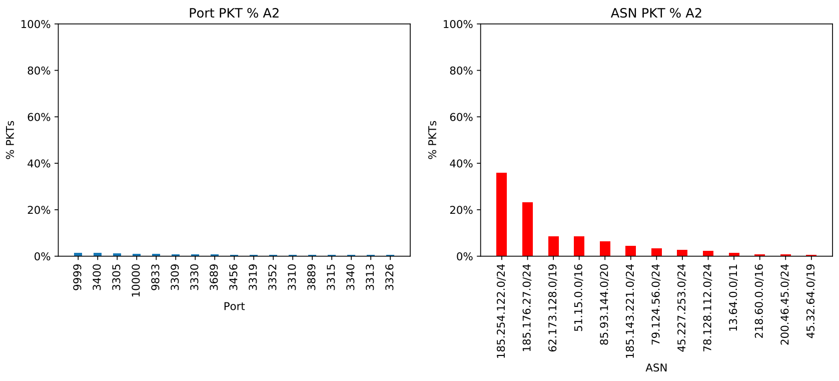


Figure 4.20: $\boldsymbol{A_2}$ similar vertical scan

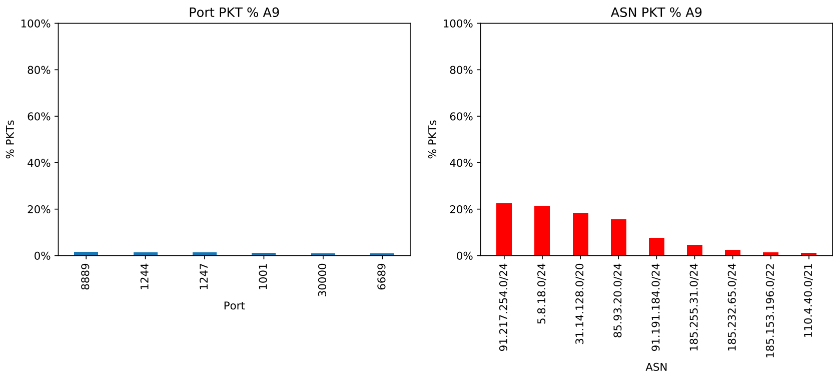


Figure 4.21: $\boldsymbol{A_9}$ similar vertical scan

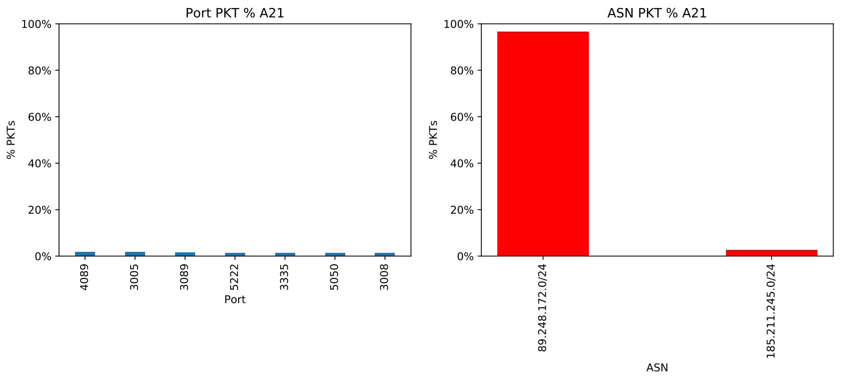


Figure 4.22: $\boldsymbol{A_{21}}$ similar vertical scan

If we look at Fig4.32 we can see that port **23** represent the 80% of destination ports labelled inside ***Port*** $\boldsymbol{A_1}$ while the IP ***IP*** $\boldsymbol{A_1}$ have a flat distribution of 1% for each value. We could say that this behaviour has a horizontal scan pattern:

```
│ 91.114.24.0/21
│     └─Port:  23
├─ 120.221.16.0/20
│     └─Port:  23
├─ 14.225.3.0/24
│     └─Port:  23
├─ 46.101.128.0/17
│     └─Port:  23
└─ ...
      └─Port:  23
```

Same conclusions can be made for $\boldsymbol{A_5, A_6, A_7, A_{10}, A_{16}, A_{17}}$:

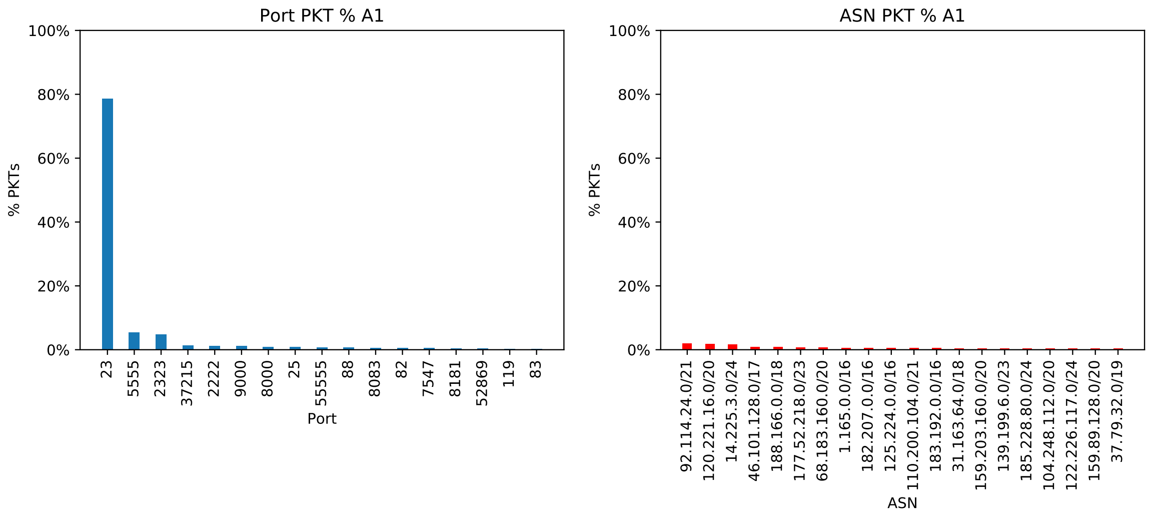


Figure 4.23: $\boldsymbol{A_1}$ horizontal scan pattern on port 23

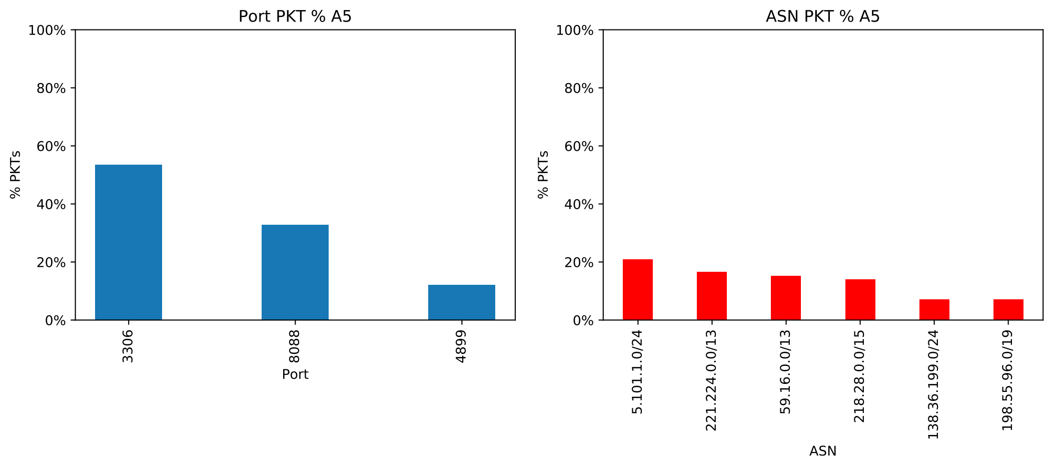


Figure 4.24: $\boldsymbol{A_5}$ Horizontal scan pattern over 3306 (MySql) et al.

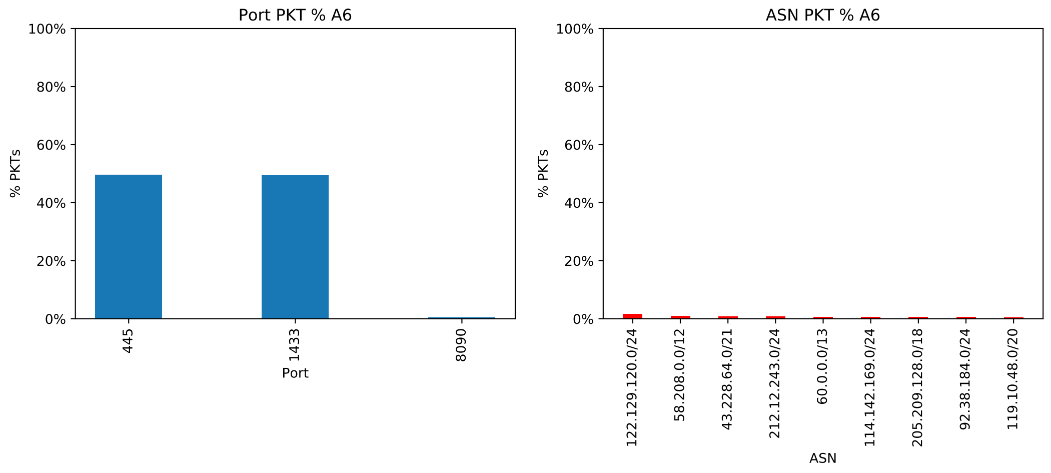


Figure 4.25: $\boldsymbol{A_6}$ Horiz. Scan on 445 and 1433 (MS-DS and MS-sql-s) Microsoft Services

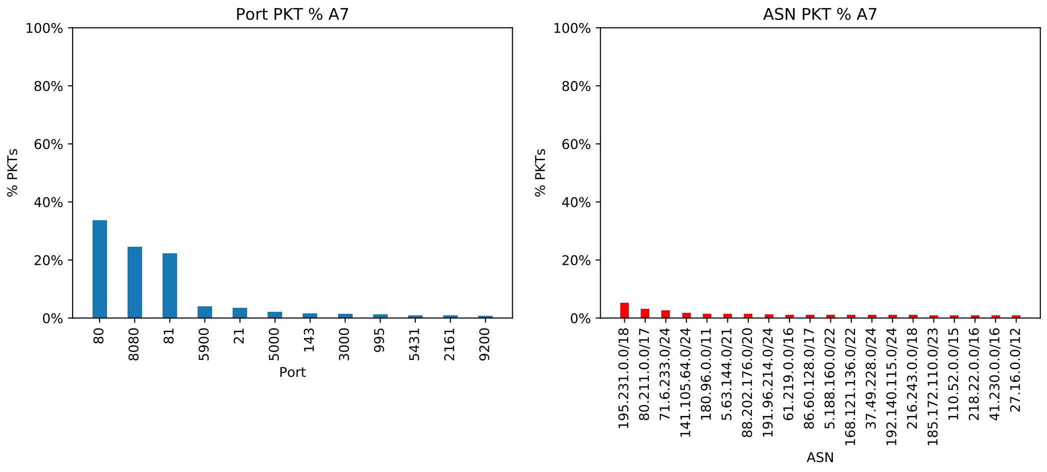


Figure 4.26: $\boldsymbol{A_7}$ HTTP ports

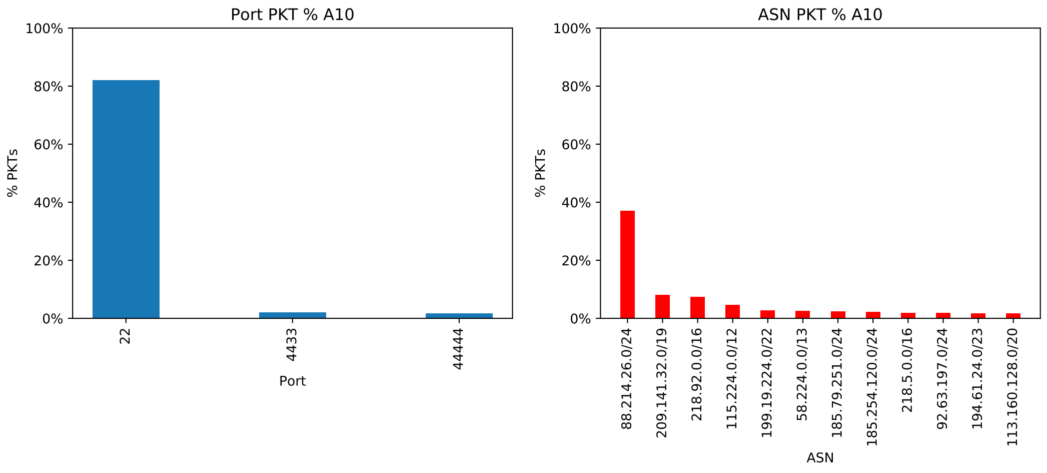


Figure 4.27: $\boldsymbol{A_{10}}$ 22 SSH Ports

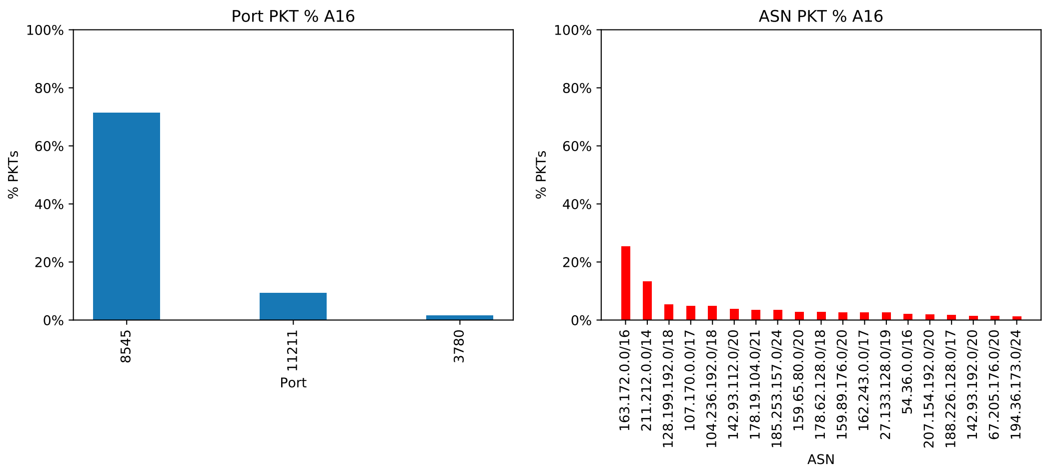


Figure 4.28: $\boldsymbol{A_{16}}$ JSON Service

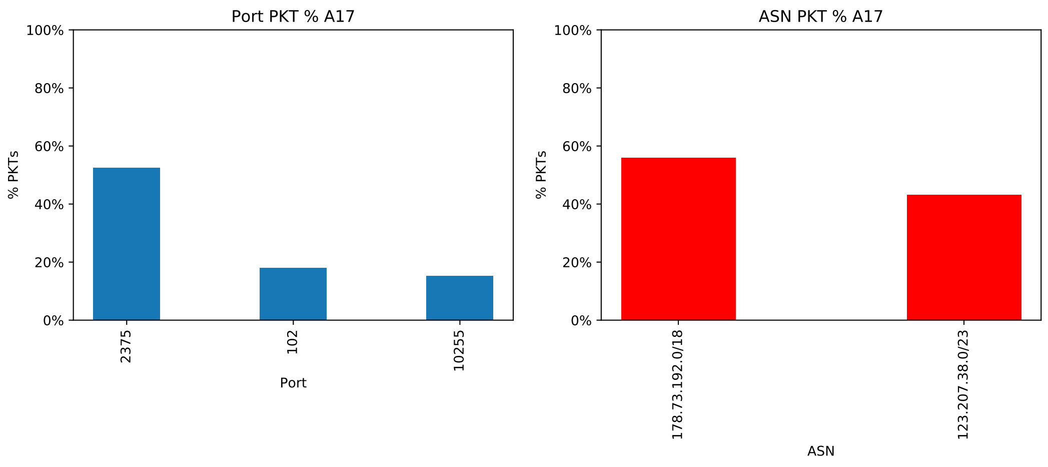


Figure 4.29: $\boldsymbol{A_{17}}$ Docker

All the other communities can be considered spurious scans or combined. We excluded from the analysis trivial communities consisting only in AS's or only ports.

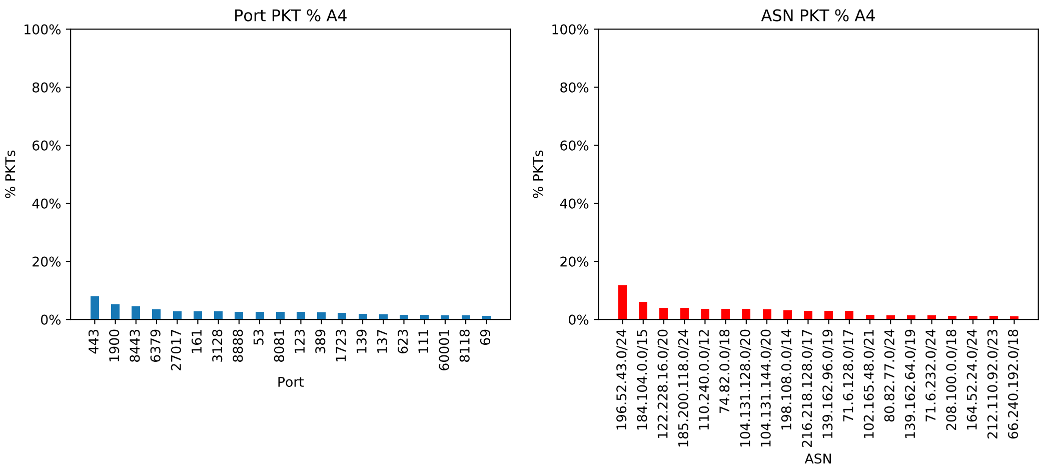


Figure 4.30: $\boldsymbol{A_4}$ Random Port-Scan with port 443 (HTTPS)

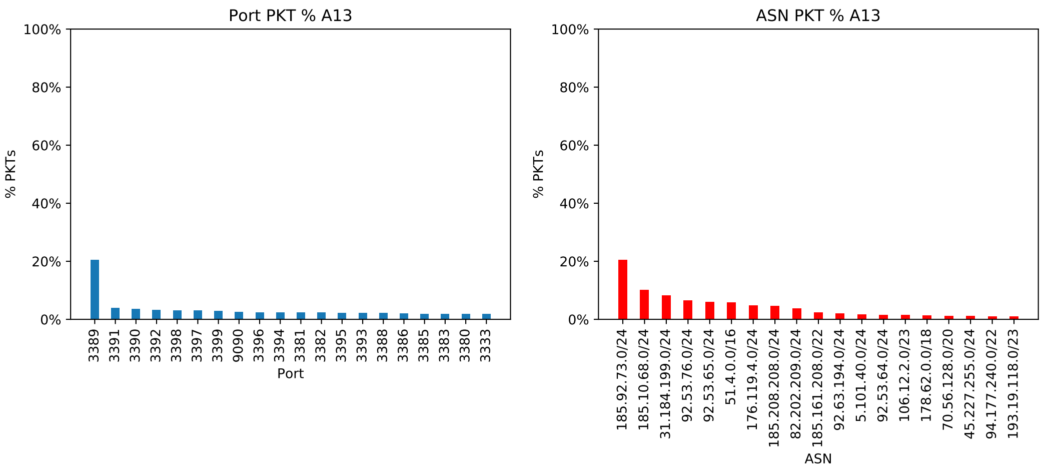


Figure 4.31: $\boldsymbol{A_{13}}$ Sequential Port Scan around port 3389 (MS-Remote Desktop)

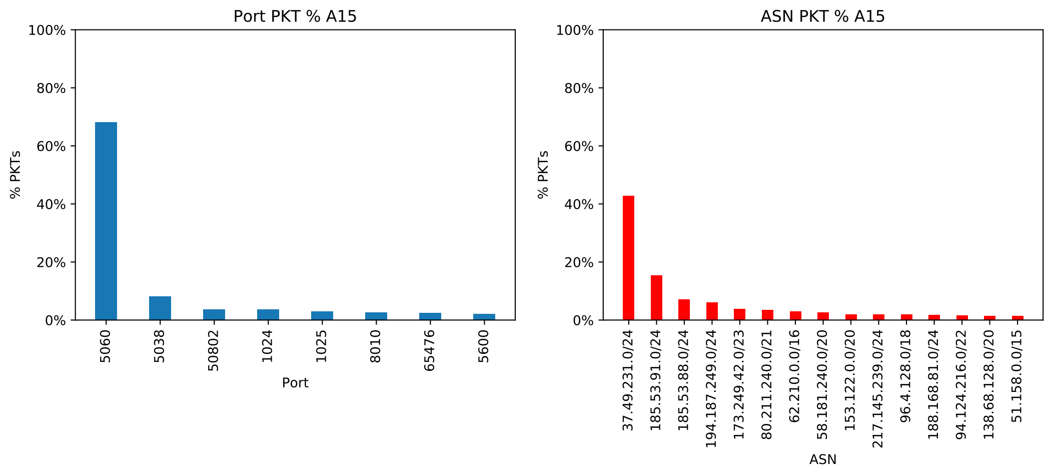


Figure 4.32: $\boldsymbol{A_{15}}$ Port 5060 SIP VoIP

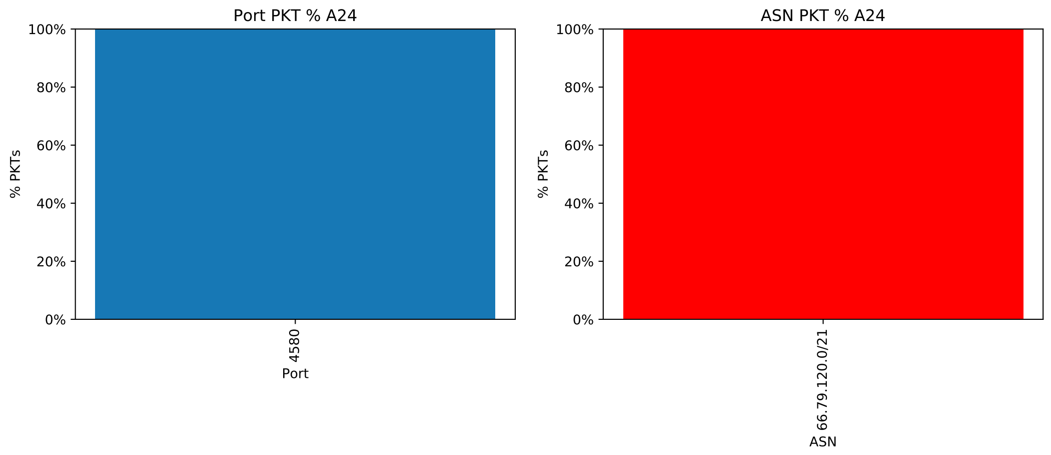


Figure 4.33: $\boldsymbol{A_{24}}$ Unassigned Port

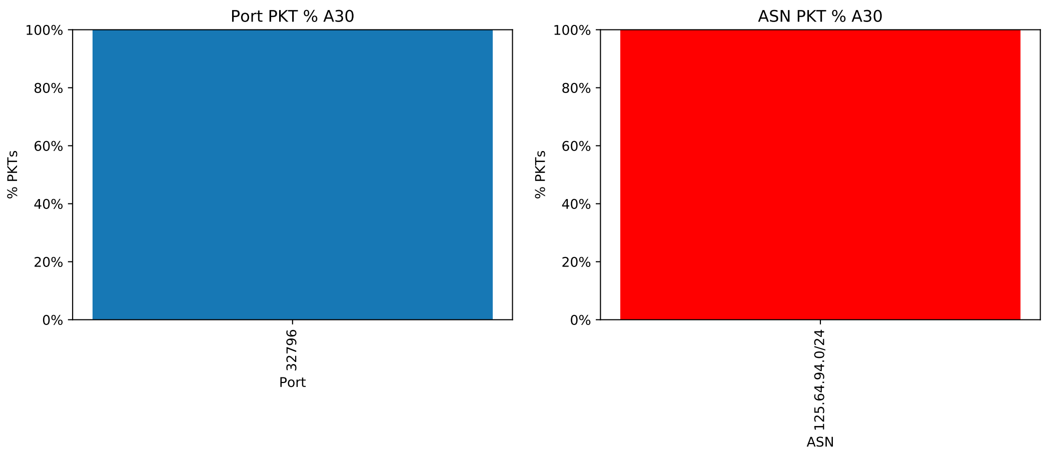


Figure 4.34: $\boldsymbol{A_{30}}$ Unassigned Port

For the analysis of February 2019 and March 2019 we will provide in this section only the figures of few illustrative examples that present a very similar behaviour with the one provided for the January 2019 analysis. For a complete view on the communities we provide all the figures in the Appendix.

4.2.3 Communities February 2019

We run the GMA[2] community detection algorithm over the sampled dataset representing the month of February 2019. The following graphs show the % of packets belonging to each community distinguished by the two labels $\boldsymbol{Port\ C_{id}}$ and $\boldsymbol{ASN\ C_{id}}$.

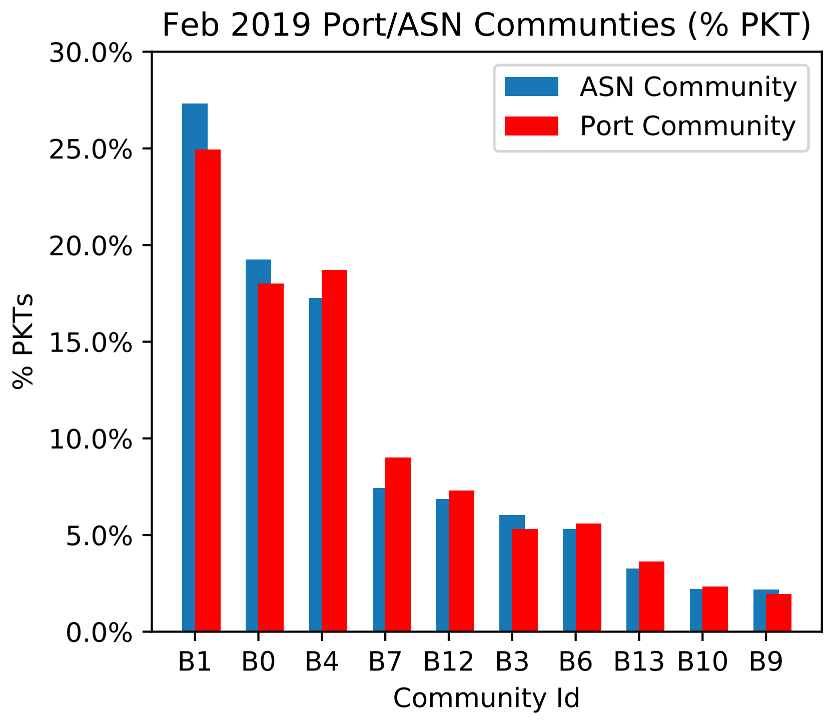


Figure 4.35: ASN/Port Communities Feb 2019

We can see from Fig.4.35 that the % of packets that present a conflict in the community identification label $\boldsymbol{C_i \not\equiv C_j}$ is not relevant. The top 3 communities wraps $\boldsymbol{\sim 70\%}$ of the filtered dataset $(\boldsymbol{B_1, B_0, B_4})$.

[2]Greedy Modularity Algorithm

Graph February 2019

The visual representation of the Community graph obtained with Gephi for February 2019 is Fig.4.18:

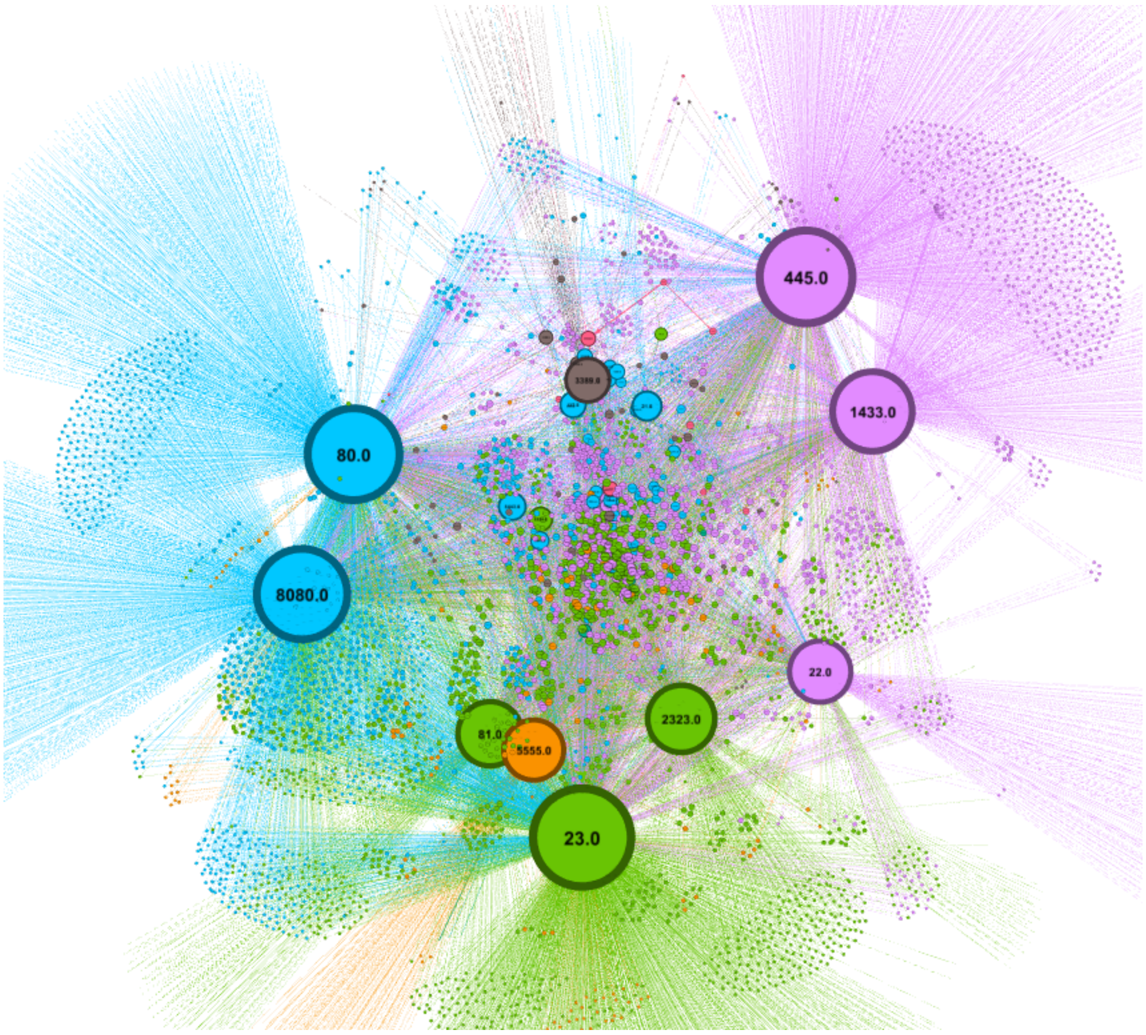


Figure 4.36: Community Graph Gephi Feb 2019

As we can se from the graph, scaled by the input-degree of the nodes, it is possible to distinguish clearly three communities, each of them consisting of source vertexes, the AS, orbiting around the centre of attraction, i.e. the port. Edges are coloured by the colour of the vertex from which they are originated, so we could say that we can distinguish again the community probing **80** and **8080**, one wrapping **23** and **81** and **2323**, another one wrapping **22 (SSH)**, **1433**, **445** ports over which usually run **Microsoft services**.

Community Analysis

Here the same premises made for Jan 2019 holds. We will directly provide communities that respect specific behavioural scan pattern.

Vertical scan communities could be $\boldsymbol{B_0, B_1, B_2, B_3, B_8, B_{11}, B_{12}}$:

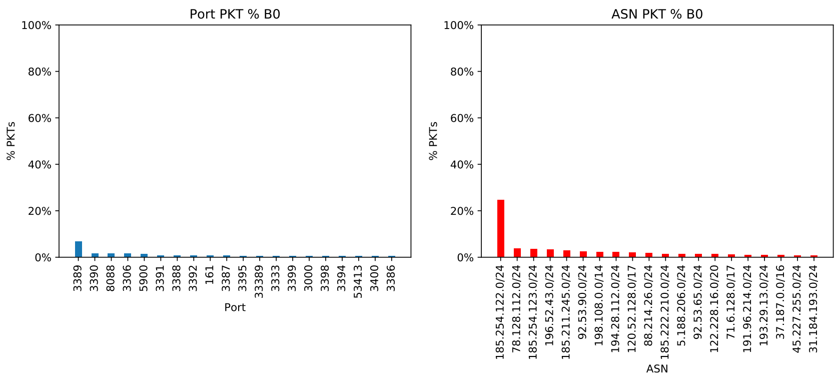


Figure 4.37: $\boldsymbol{B_0}$

All the related figures are provided in Appendix. Horizontal scan communities could be $\boldsymbol{B_4, B_9, B_7, B_{13}}$:

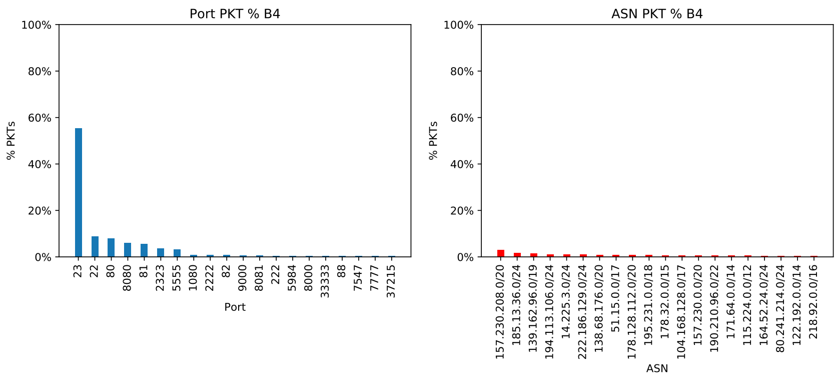


Figure 4.38: $\boldsymbol{B_4}$

All the related figures are provided in Appendix.

All the other communities can be considered spurious scans or combined and the figure are provided in appendix. We excluded from the analysis trivial communities consisting only in AS's or only ports.

Communities March 2019

We run the GMA[3] community detection algorithm over the sampled dataset representing the month of March 2019. The following graphs show the % of packets belonging to each community distinguished by the two labels $\boldsymbol{Port\ C_{id}}$ and $\boldsymbol{ASN\ C_{id}}$.

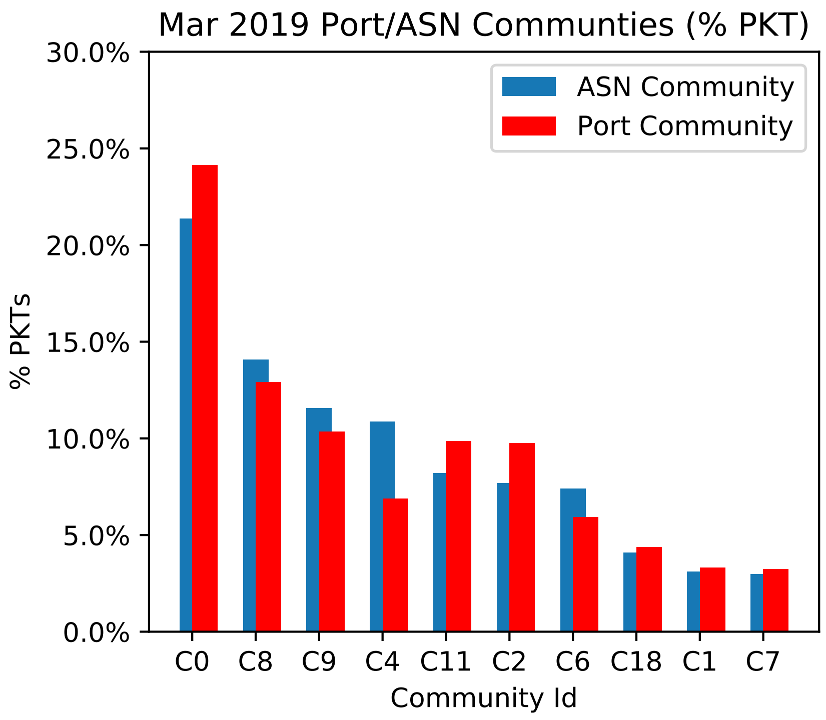


Figure 4.39: ASN/Port Communities March 2019

We can see from Fig.4.39 that the % of packets that present a conflict in the community identification label $\boldsymbol{C_i} \not\equiv \boldsymbol{C_j}$ is not relevant. As we will se from the next section we can already highlight three communities grouping 50% of the traffic $(\boldsymbol{C_0, C_8, C_9})$.

[3]Greedy Modularity Algorithm

Graph March 2019

The visual representation of the Community graph obtained with Gephi for March 2019 is Fig.4.18:

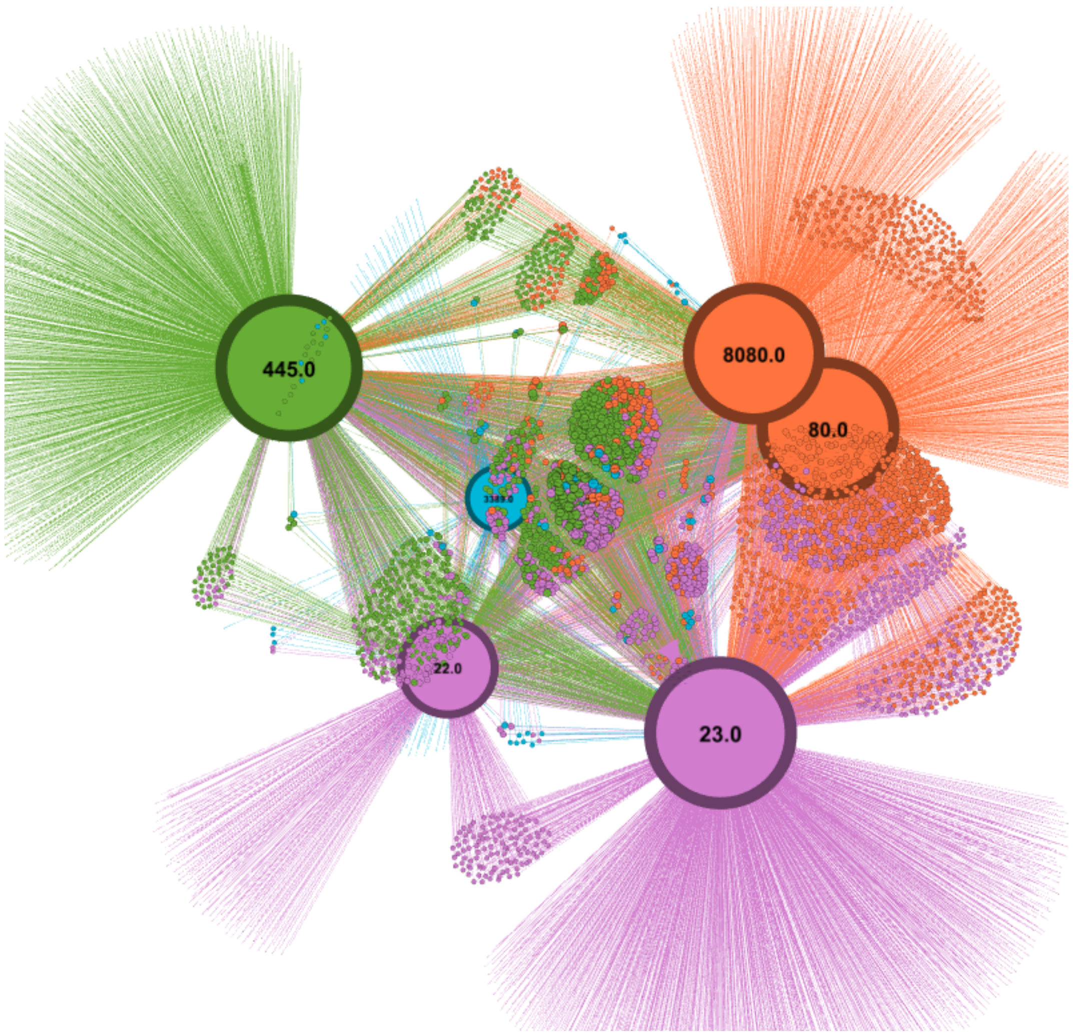


Figure 4.40: Community Graph Gephi Mar 2019

We can highlight four communities: one orbiting around **455**, the second centered on **80** and **8080**, a small cluster around **3389** and another important community around ports **23** and **22**

Community Analysis

Here the same premises made for Jan 2019 holds. We will directly provide communities that respect specific behavioural scan pattern.

Vertical scan communities could be $\boldsymbol{C_5, C_7, C_8, C_9, C_{10}, C_{12}, C_{16}, C_{18}, C_{21}}$:

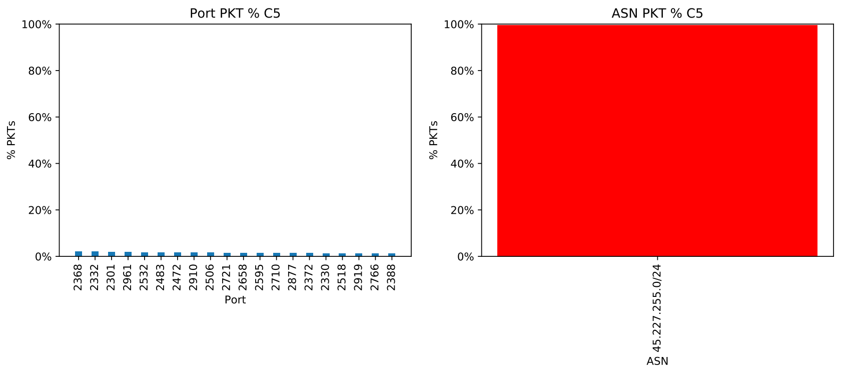


Figure 4.41: $\boldsymbol{C_5}$

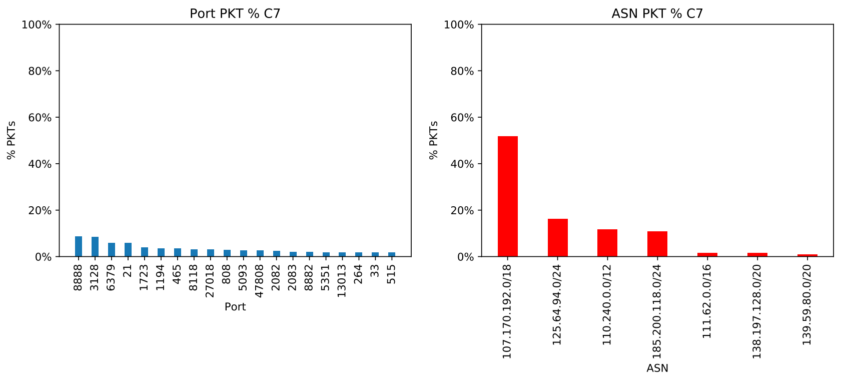


Figure 4.42: $\boldsymbol{C_7}$

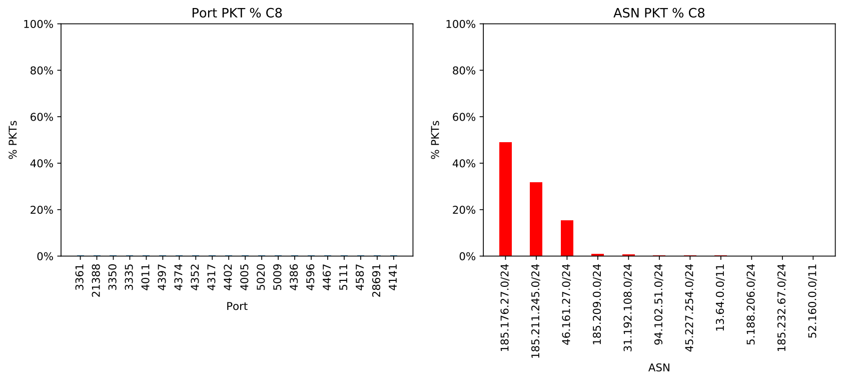


Figure 4.43: $\boldsymbol{C_8}$

All the remaining communities figures are provided in the Appendix.

Horizontal scan communities could be $\boldsymbol{C_0, C_1, C_2, C_3, C_4}$:

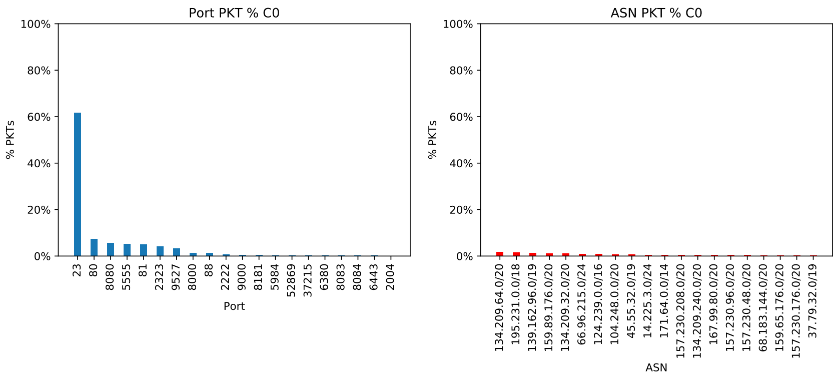


Figure 4.44: $\boldsymbol{C_0}$ Horizontal Scan on port 23 (Telnet)

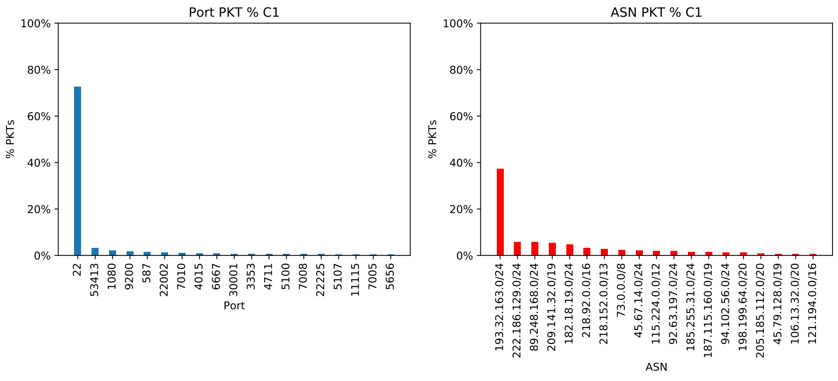


Figure 4.45: $\boldsymbol{C_1}$ Horizontal Scan on port 22 (SSH)

All the other communities can be considered spurious scans or combined. As already mentioned these figures will be provided in the Appendix. We excluded from the analysis trivial communities consisting only in AS's or only ports.

4.3 Communities Jaccard Similarity

The second step of the $\boldsymbol{ASN} \rightarrow \boldsymbol{P}$ community detection analysis is to highlight the similarities during time between the three dataset clusters. The previous results already highlighted at a glance clear similarities that we will now present through a the numerical distance metric represented by the Jaccard Distance metric score as defined in the 2.4. Also here we will make distinction between sets of $\boldsymbol{Port\ C_{id}}$ and $\boldsymbol{ASN\ C_{id}}$. For sake of simplicity we transformed the symmetric matrix into a upper right triangular one where the left diagonal is unitary since each $\boldsymbol{C_0}$ has a score of 1.0 similarity to itself.

4.3.1 ASN Jaccard Matrix

The first result provided compares the similarity score obtained between the three months computed over the set of ASN present for each the community $\boldsymbol{ASN\ A_{id}, B_{id}C_{id}}$. The colour bar highlights the groups that present the highest intersection.

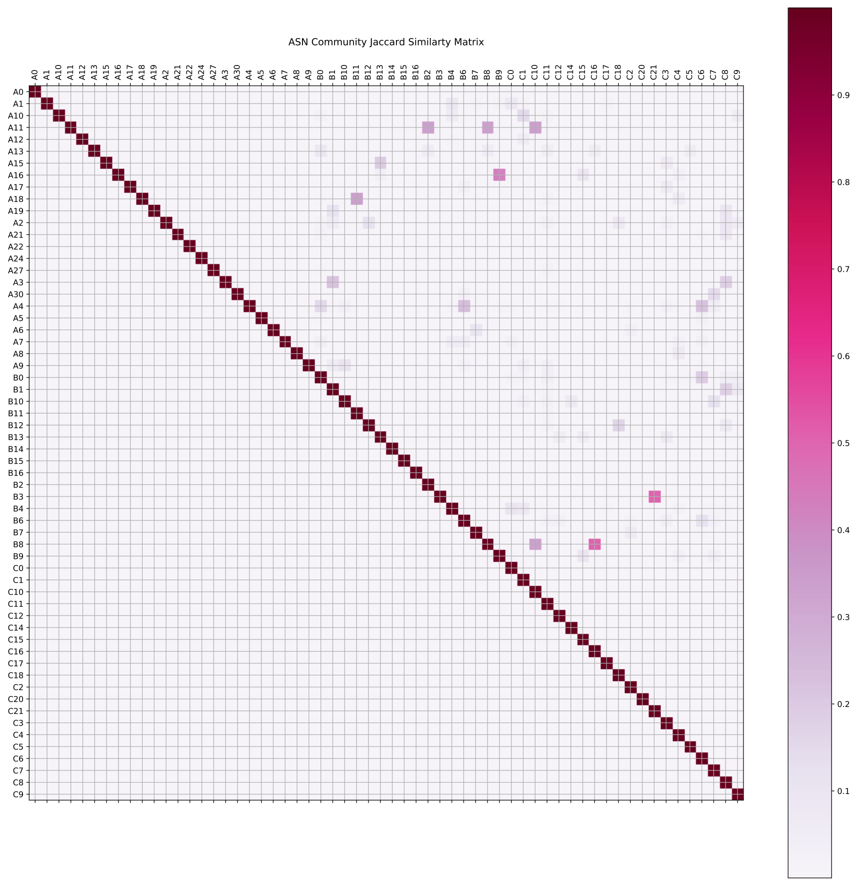


Figure 4.46: ASN Jaccard Matrix

In order to make the understanding more clear we will decompose the matrix in three sub-matrices which compare between combinations of month. Let's look at the comparison between January 2019 and February 2019. The Jaccard score between the sets of ASN does not present results that overcome the threshold of 0.5 similarity.

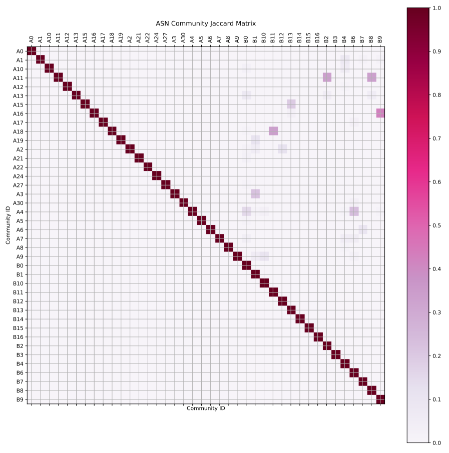


Figure 4.47: ASN Jaccard Matrix Jan/Feb

Among the most similar communities we can point out

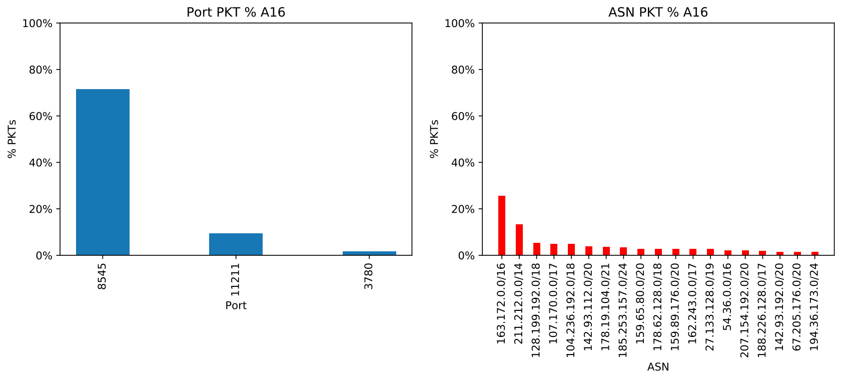


Figure 4.48: $\boldsymbol{A_{16}}$

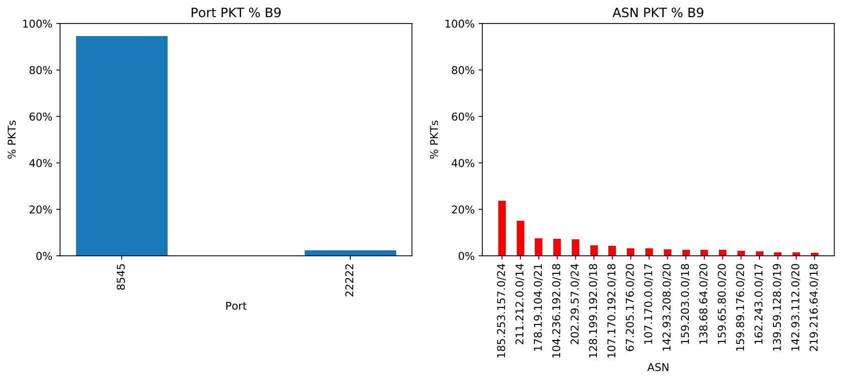


Figure 4.49: $\boldsymbol{B_9}$

Let's look at the comparison between February 2019 and March 2019 . The Jaccard score between set of port does not present results that overcome the threshold of 0.7 similarity:

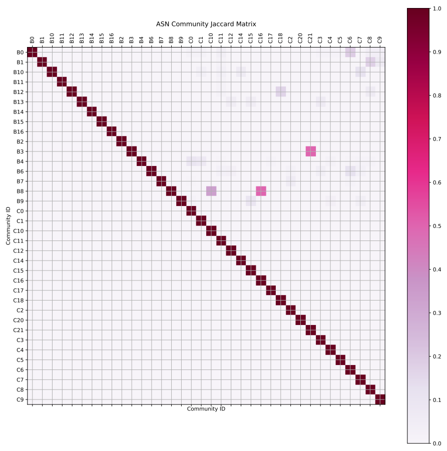


Figure 4.50: ASN Jaccard Matrix Jan/Mar

Among the most similar communities we can point out

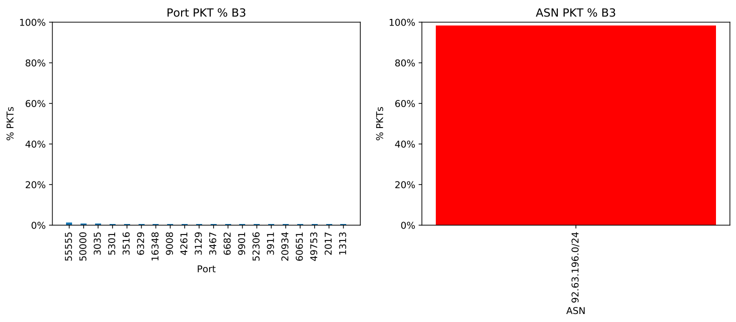


Figure 4.51: $\boldsymbol{B_3}$

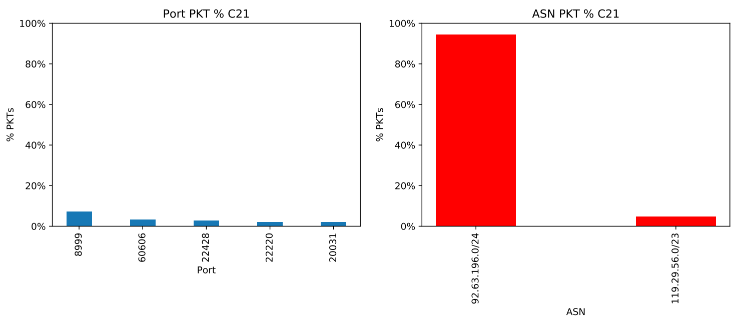


Figure 4.52: $\boldsymbol{C_{21}}$

The comparison between January and March 2019 is omitted since no relevant similarity was observed looking at the AS set.

4.3.2 Port Similarities

The first result provided compares the similarity score obtained between the three months computed over the set of Ports present for each the community $\boldsymbol{Port\ A_{id}, B_{id}C_{id}}$. The colour bar highlights the groups that present the highest intersection.

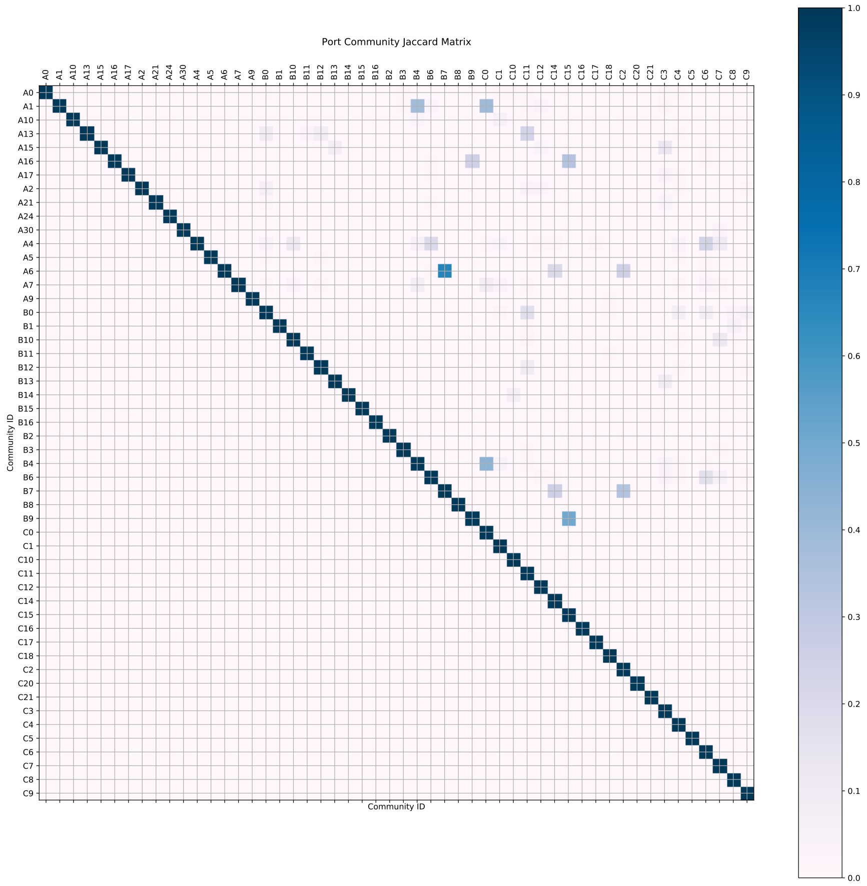


Figure 4.53: Port Jaccard Matrix

Let's look at the comparison between January 2019 and February 2019. The Jaccard score between the sets of ports present two close couple of communities:

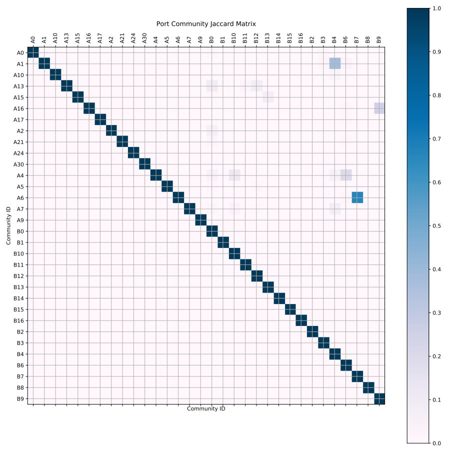


Figure 4.54: Port Jaccard Matrix Jan/Feb

$\boldsymbol{A_6}$ and $\boldsymbol{B_7}$

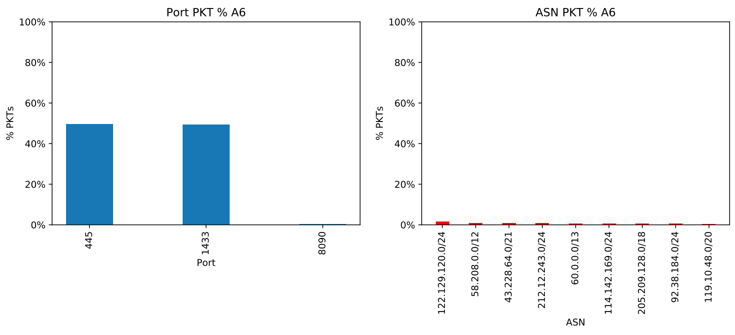


Figure 4.55: $\boldsymbol{A_6}$

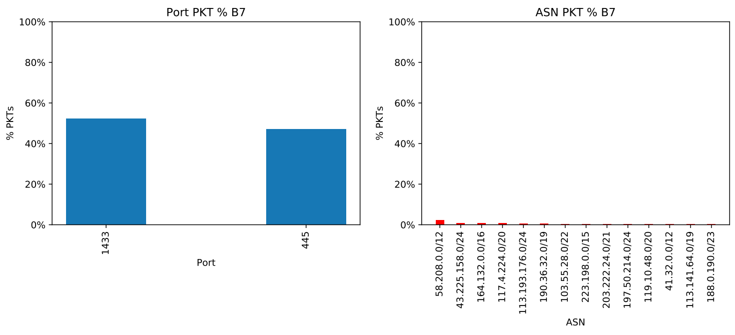


Figure 4.56: $\boldsymbol{B_7}$

$\boldsymbol{A_1}$ and $\boldsymbol{B_4}$

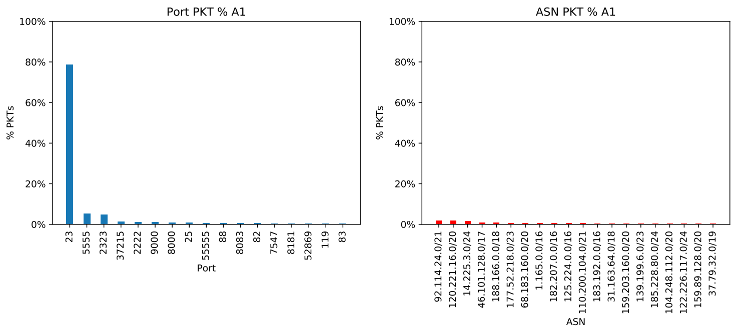


Figure 4.57: $\boldsymbol{A_1}$

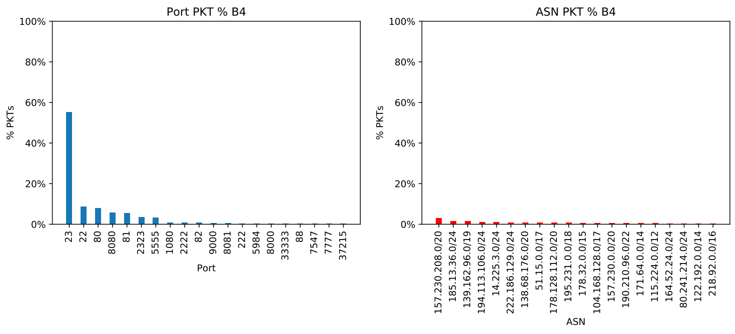


Figure 4.58: $\boldsymbol{B_4}$

Let's look at the comparison between January 2019 and March 2019. The Jaccard score between the sets of ports present few similar clusters, with an highest score of 0.4.

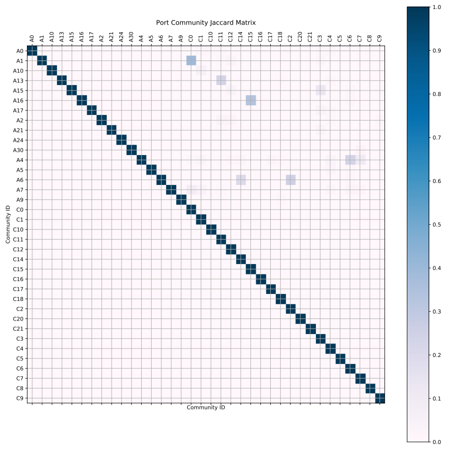


Figure 4.59: Port Jaccard Matrix Jan/Mar

$\boldsymbol{A_1}$ and $\boldsymbol{C_0}$

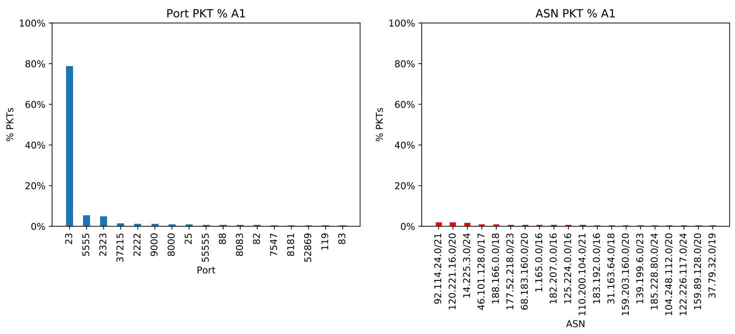


Figure 4.60: $\boldsymbol{A_1}$

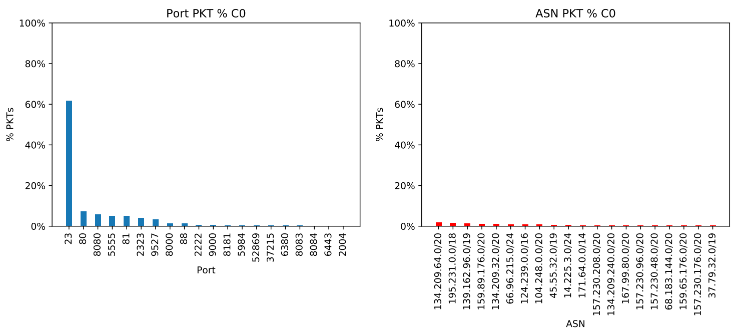


Figure 4.61: $\boldsymbol{C_0}$

$\boldsymbol{A_{16}}$ and $\boldsymbol{C_{15}}$

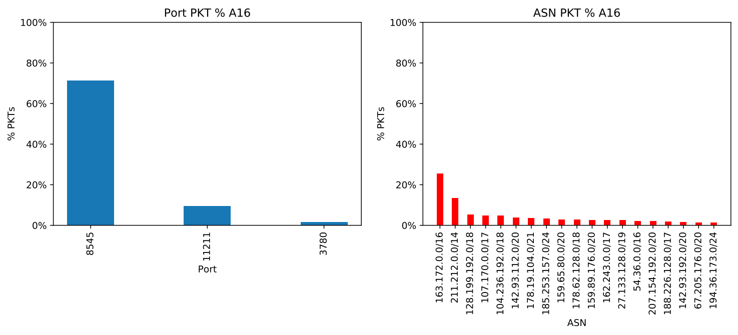


Figure 4.62: $\boldsymbol{A_{16}}$

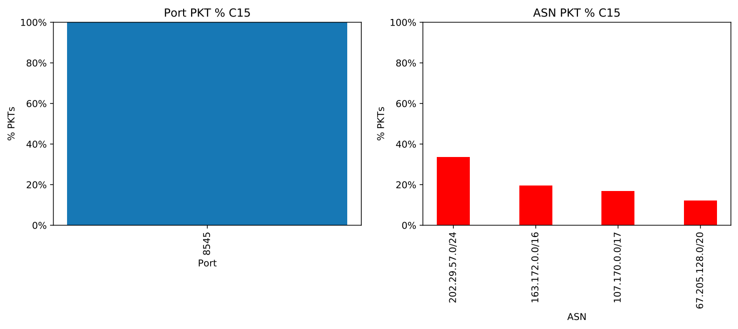


Figure 4.63: $\boldsymbol{C_{15}}$

Let's look at the comparison between February 2019 and March 2019. The Jaccard score between the sets of ports present few similar clusters. More precisely:

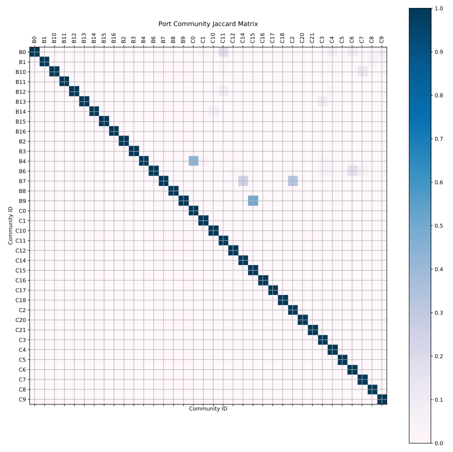


Figure 4.64: Port Jaccard Matrix Feb/Mar

$\boldsymbol{B_4}$ and $\boldsymbol{C_0}$:

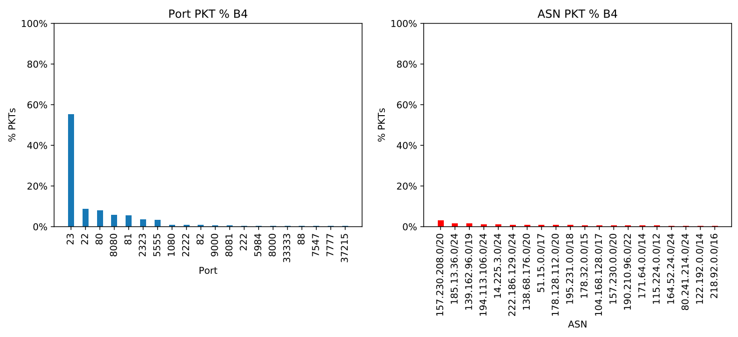


Figure 4.65: $\boldsymbol{B_4}$

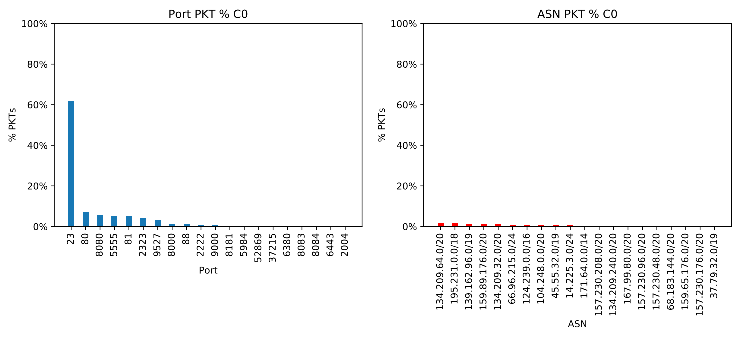


Figure 4.66: $\boldsymbol{C_0}$

B_9 and C_{15}:

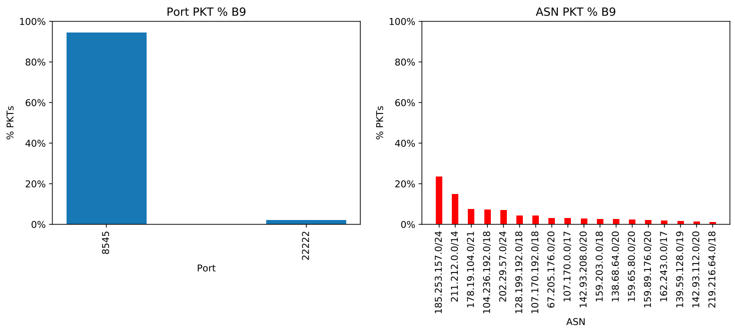


Figure 4.67: $\boldsymbol{B_9}$

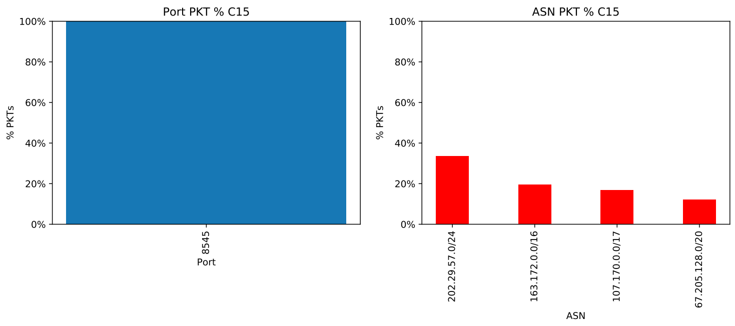


Figure 4.68: $\boldsymbol{C_{15}}$

4.4 Port Scan Graph

After running GMA on Gephi over the port sequences graph we obtained the following results:

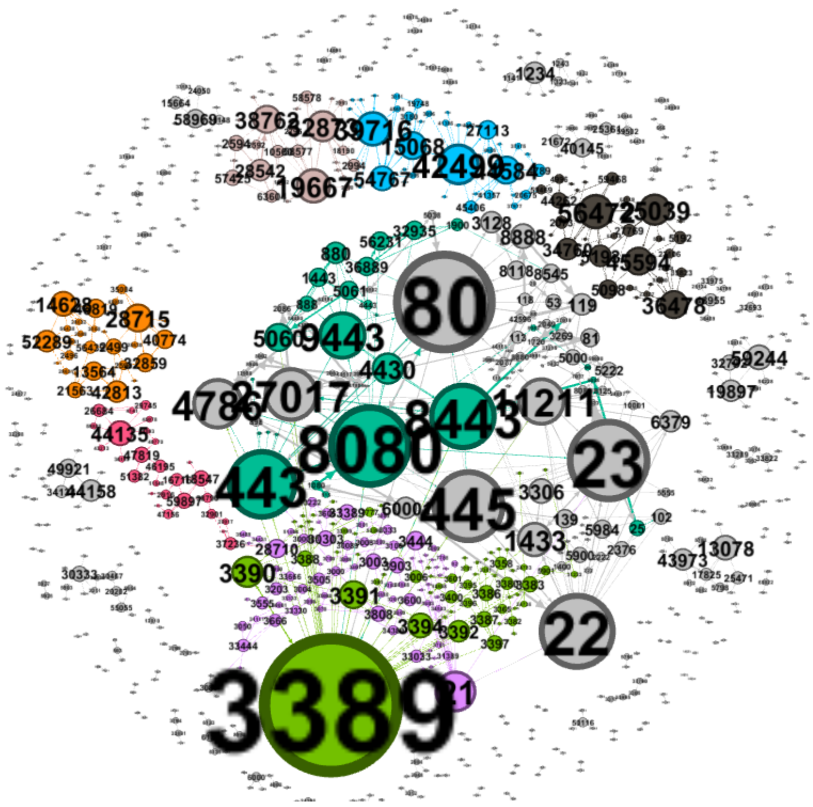


Figure 4.69: Port Scan Graph January 2019

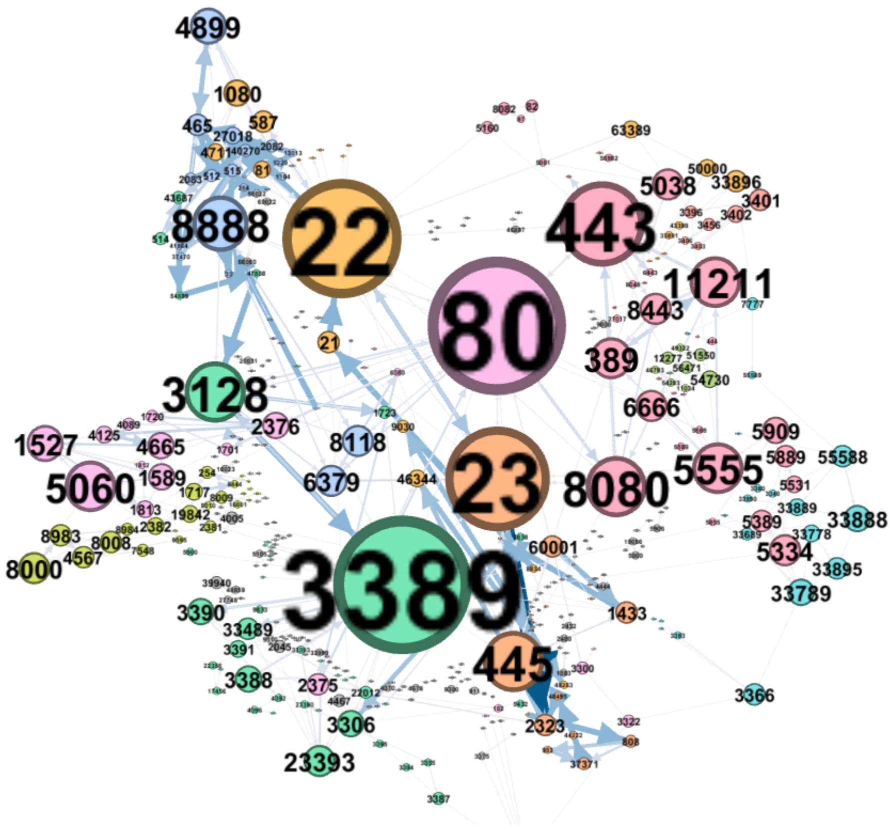


Figure 4.70: Port Scan Graph February 2019

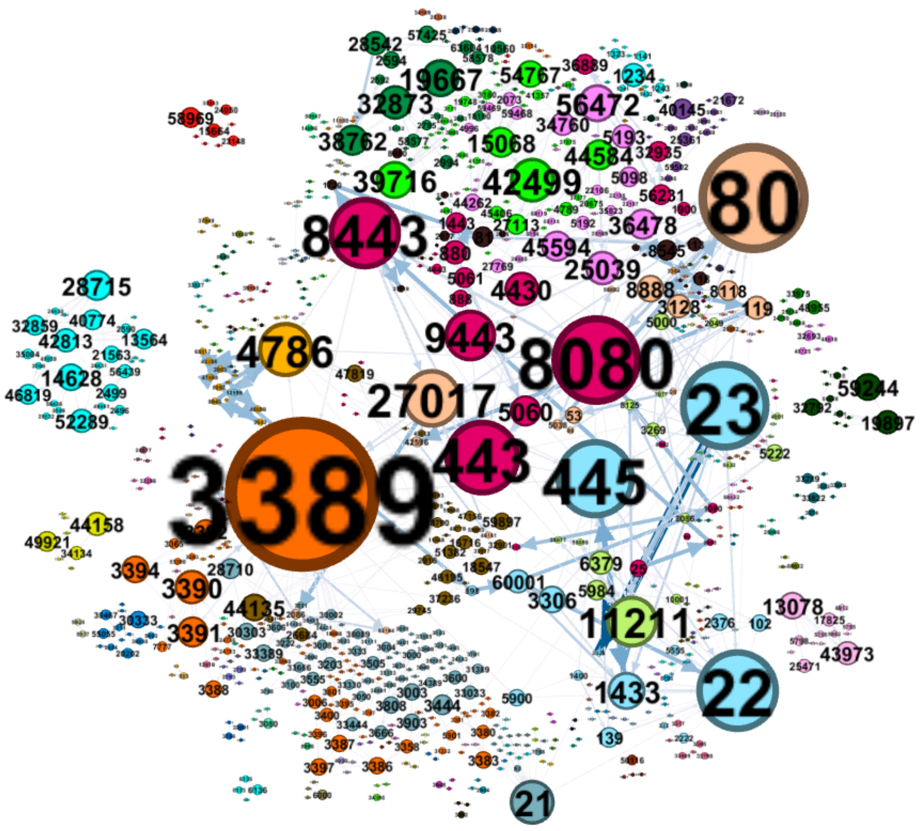


Figure 4.71: Port Scan Graph March 2019

The graphs highlight communities of consecutive port scanning that follows sequential numbering in order to cover the interested ports. We can see a strong relationship between port 23 and 2323, same can be said for port in range around 3389 and 40000. We can see that all the three time window highlights the frequency of specific port patterns that does not change over time giving support to the assumptions that we would expect communities that consists of particular sequential or random pattern of port scanning which wraps the well-know port over which popular internet application are provided and popular vulnerabilities can be exploited. Furthermore we confirm that this approach that has been already developed in previous studies can cover the part of the spurious traffic that targets "noisy" or "unpopular" ports.

Chapter 5

Conclusions

The goal of this work was to **dig into the spurious Darknet traffic** in order to isolate groups of **"dirty ASes"**.

We made use of community detection algorithm, assuming that the suspicious activities could be well represented by a "social relationship" graph between ASes and ports.

We obtained **results that highlighted the presence of some big groups of ASes** performing **similar type of activities** (vertical scans, targeted scans over popular services etc.). This outcomes were consistent also thanks to the preprocessing phase in which we characterized the traffic and pruned the rows that we did not considered relevant. We lighten the size of the analysed capture, using a simple graph definition and an existing heuristic community detection algorithm GMA without losing consistency on the results. Another important goal was to check whether the dirty sources were repetitive over time. We found those traces proving that this spurious **actors can be recurrent over time** thanks to the creation of Jaccard Similarity distance matrix. Not only we highlighted traces of similar communties of ASes but also found some little evidence that also the targeted port were recurrent.

Checking communities of port-scan sequences we took into account the noise filtered in the main analysis and confirmed results already obtained in previous similar works, obtaining communities of sequential and random scan sequences.

The darknet proved to be powerful tool in order to tailor the research on the features of the spurious traffic that otherwise would be lost in the large amount of the legitimate users. The work tried to prove that consistent initial cyber-intelligence results can be obtained with a simple, non computationally expensive, and light approach. Eventually this outcomes can be used to profile ASes and eventually find malicious traffic in operational network. This work can also provide initial knowledge for active machine learning approaches.

Ringraziamenti

Prima di tuffarmi in una scrittura poco ortodossa, voglio formalmente ringraziare il mio relatore **Prof. Marco Mellia** e il mio correlatore D**r. Idilio Drago** per il modo in cui mi hanno saputo trasmettere sicurezza e tranquillità, per il supporto umano e materiale prestato durante il periodo di tesi.
Contrariamente ai precedenti periodi di ribellione, mi sono concesso il lusso di scrivere dei ringraziamenti. Sarò prolisso e poco formale, sarà una rara occasione di poter venire a contatto con la mia parte più interna, più che un ringraziamento sarà una confidenza che spero il lettore comprenderà con sensibilità ora e in futuro.
Inizio naturalmente ringraziando i miei genitori **Mariella** e **Filippo** per essere stati un argine alle mie frequenti esuberanze ed eruzioni, un conforto nei momenti di difficoltà sopratutto umani, che sono stati tanti e sappiamo che lo saranno ancora: grazie mamma per la tua pazienza e sensibilità, grazie papà per la tua determinazione e pragmaticità, per il vostro non trascurabile sostegno materiale che spesso viene dato per scontato troppo facilmente. Grazie a mia sorella **Roberta** per le prese per i fondelli, per i momenti di leggerezza, e per l'affetto, nonostante i periodi di assenza e i miei lunghi silenzi. Un posto speciale è naturalmente riservato anche ai miei nonni. **Nonna Tonina** grazie per il tuo spropositato affetto incondizionato e per la tua dolcezza, efficacemente rappresentati dalla bontà e dalla dimensione dei tuoi piatti che mi vengono in sogno frequentemente in fase di astinenza, grazie **nonno Mauro** che non ho mai conosciuto ma di cui tengo orgogliosamente vivo il nome; grazie **nonna Maria** per la tua tenacia e la tua cucina, grazie **nonno Giuseppe** per la tua pacatezza e la tua ironia, per il tuo amore per la terra che spero di imparare: quando ritrovo tutti e tre dopo le mie lunghe mancanze pare non sia passato neanche un giorno dal nostro ultimo incontro. In ultimo ringrazio **zie**, **zii**, **cugini** e **cugine**, e chiunque altro orbiti attorno a questo nucleo familiare, per l'impronta lasciata su di me, un modello da cui trarre insegnamenti positivi ma anche di cui correggerne gli errori.

Passo ora a tessere le lodi e i ringraziamenti dei miei amici, la cui priorità è prevedibilmente data a **Maurizio e Gilberto**, **fratelli** acquisiti, amici decennali, in perenne conflitto e confronto interno, veri, veraci, simili e complementari a me stesso. Il tempo e i chilometri ne certificano l'autenticità, un porto sicuro in questi due anni. Grazie **Domenico**, nonstante il tuo essere schivo, riservato e competitivo, per l'amicizia ventennale, lo spirito critico, la sensibilità. Grazie in ordine sparso a **Corrado** per i simposi sul dialetto, sport, politica, cibo; grazie **Pierangelo** per i frequenti passaggi; **Chiara M., Chiara C., Alessandra C.** per la loro diversa e femminile capacità di ascoltare; **Danilo, Giovanni, Maurizio d.B., Cesare, Paola, Alessandra V., Maria, Silvana, Brigida**, etc. Grazie a tutti per le vostre peculiarità, stravaganze, qualità, difetti malcelati, estri artistici. Ringrazio anche **Paolo, Luca, Mister Rana, Vincenzo** e gli altri amici di scuola. **Donato** che avrebbe meritato glorie in passato in quanto compagno di banco fra liceo e triennale, un'avventura di cui andare fieri. Ruffianamente aggiungerei un grazie a chiunque senta il merito di dover apparire in questa lista per avere ricevuto e offerto anche un briciolo di affetto, da e per me, che ho omesso non volutamente. Grazie a chi ha condiviso con me le strade di Torino, nelle poche uscite che ho concesso: **Giulia** paziente, positiva e disponibile a confrontarsi con le mie tristezze da emigrante; **Vincenzo**, **Camillo**, **Giovanni**, **Andrea**, **Rizvan** commilitoni di leva qui a Torino, con cui ho affrontato difficoltà accademiche, diete fallite, gelati e sushi, e i poco fruttiferi e rari allenamenti in palestra. Un grazie al **Politecnico di Torino** e alla **città** che, nonostante io sia refrattario ad ammetterlo, mi hanno quasi sempre concesso tutto ciò che mi sarei aspettato alla partenza. Ringrazio tutti i miei amici internazionali in particolare **Diego**, **Jorge**, **Sam**. Ringrazio **Soňa** per tutto ciò che è stato.

Al termine di questo percorso educativo non posso omettere dai ringraziamenti colei che in origine ha visto alberi li dove c'erano ancora germogli, aprendo la mia mente, la **maestra Tina**: una piccola parte di questo percorso è frutto di un seme che tu hai iniziato a coltivare. C'è ancora spazio per non trascurare la **Prof. Bandini**, che per prima mi ha fornito un esempio dell'importanza della disciplina e della costanza necessarie per lo studio delle metalliche materie scientifiche, attirandosi lodi e inevitabili critiche.

Ora che la lettura inizia ad essere pesante, e le dimenticanze inizieranno ad essere gravi, lancio gli ultimi ringraziamenti. Ringrazio il **Politecnico di Bari** per quello che mi ha dato nel bene e nel male durante la mia triennale. Grazie alla mia terra, la **Puglia**, che amo e per cui mi danno in egual misura. Ad esempio a me piace il panorama dalla mia camera, il Castel del Monte sulla linea dell'orizzonte, i cieli azzurri, il calore dei pomeriggi, la freschezza delle notti insonni estive, mi piace scoprire lontano il mare, il cielo all'imbrunire, seguire la luce delle lampare. Una serie di banalità che non potrei però tralasciare: se mai qualcuno capirà sarà senz'altro un altro come me. Per tutti questi motivi provo a correre con l'idea forse impopolare che la meta era già stata tracciata sin dall'inzio e coincide col punto di partenza: casa, di cui porterò sempre con me tutto il meglio e tutto il peggio. I mie sentimenti sfuggono alle briglie della mia razionalità, mi perdonerete per questa insuale e momentanea perdita di controllo.

Grazie!

Appendix A

Appendix

A.1 Communties February 2019

A.1.1 Vertical Scan Feb 2019

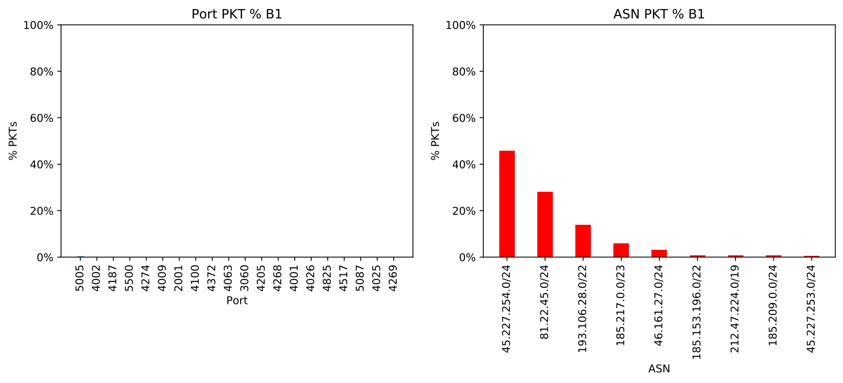


Figure A.1: ***B*$_1$**

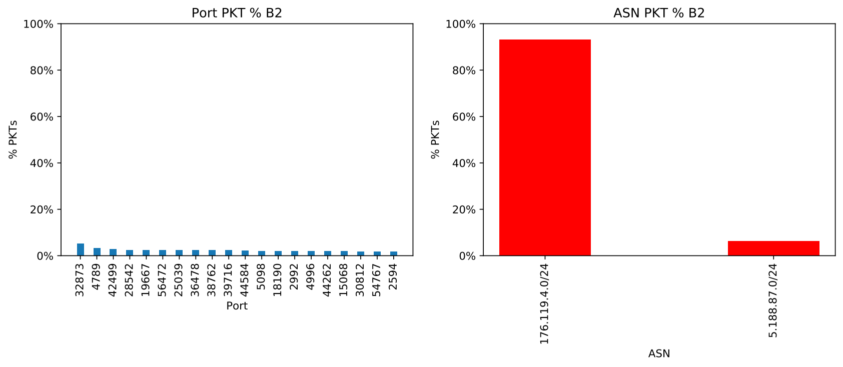


Figure A.2: ***B*$_2$**

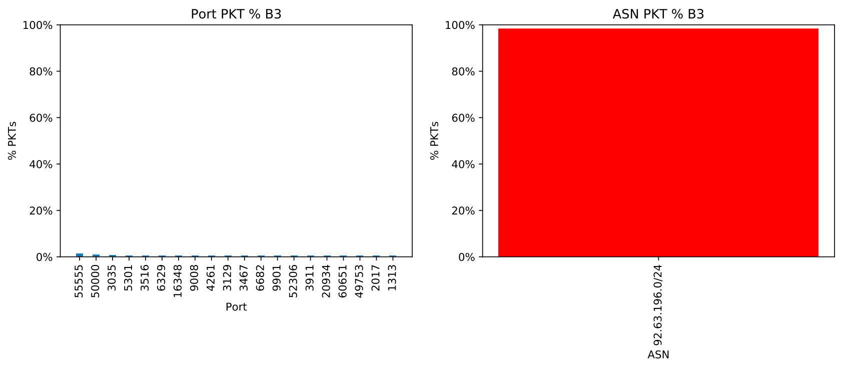


Figure A.3: $\boldsymbol{B_3}$

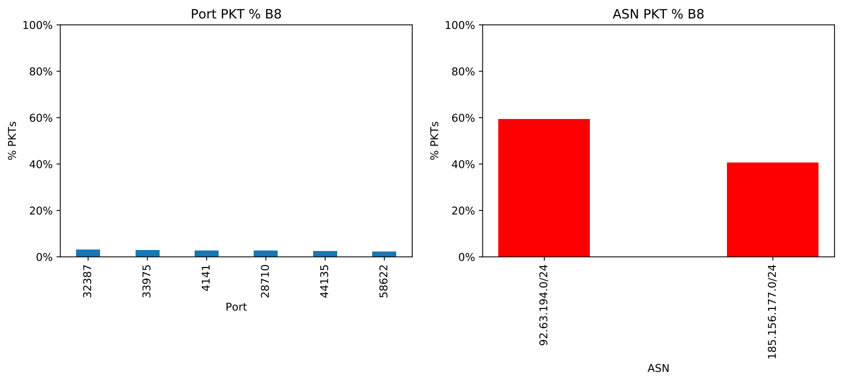


Figure A.4: $\boldsymbol{B_8}$

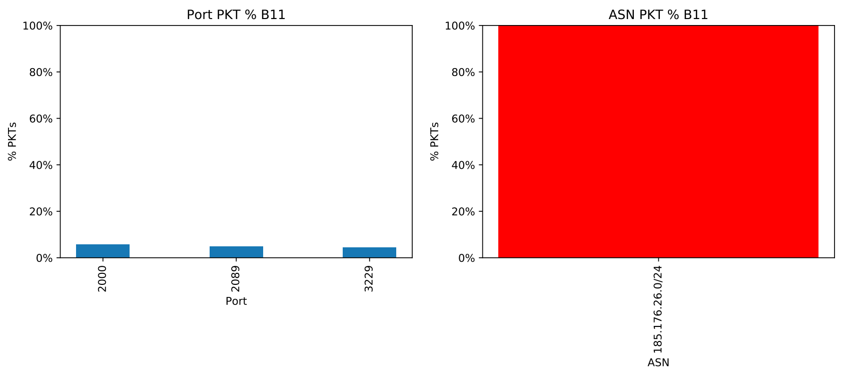


Figure A.5: $\boldsymbol{B_{11}}$

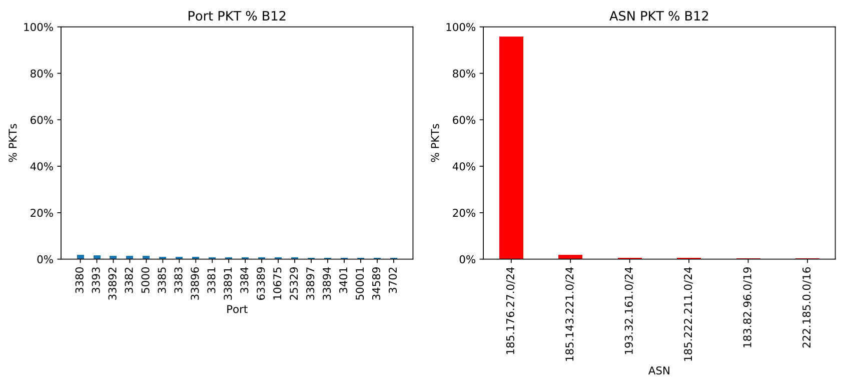


Figure A.6: $\boldsymbol{B_{12}}$

A.1.2 Horizontal Scan Feb 2019

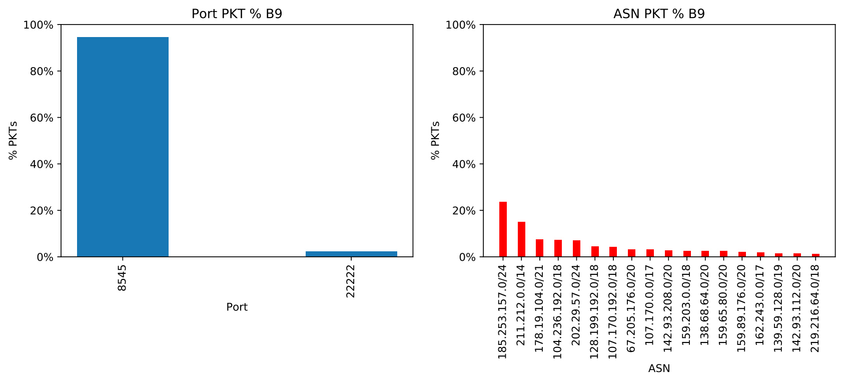


Figure A.7: $\boldsymbol{B_9}$

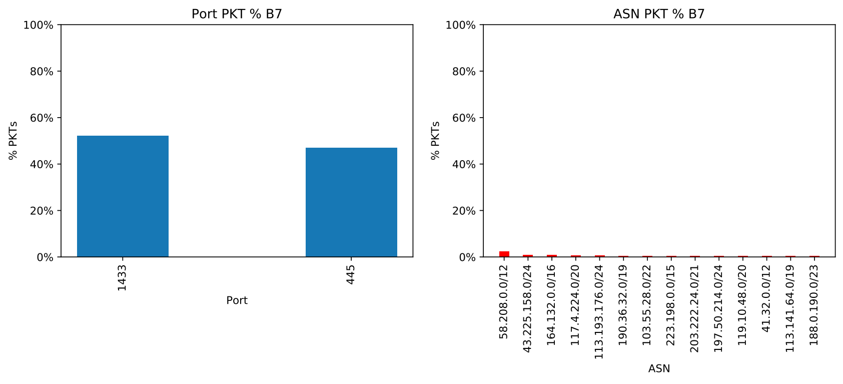


Figure A.8: $\boldsymbol{B_7}$

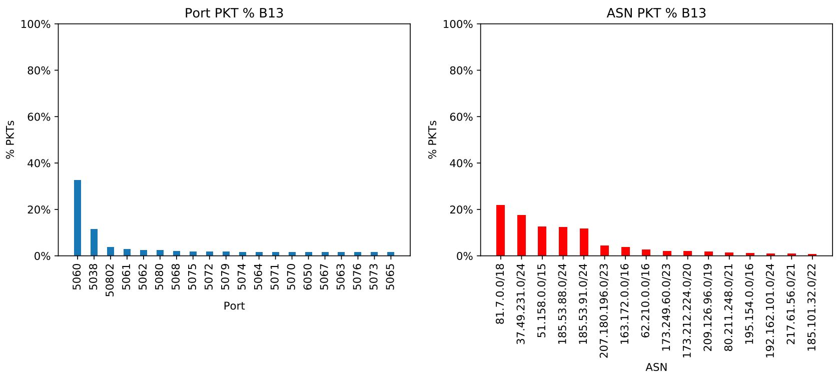


Figure A.9: $\boldsymbol{B_{13}}$

A.1.3 Miscellaneous Scan Feb 2019

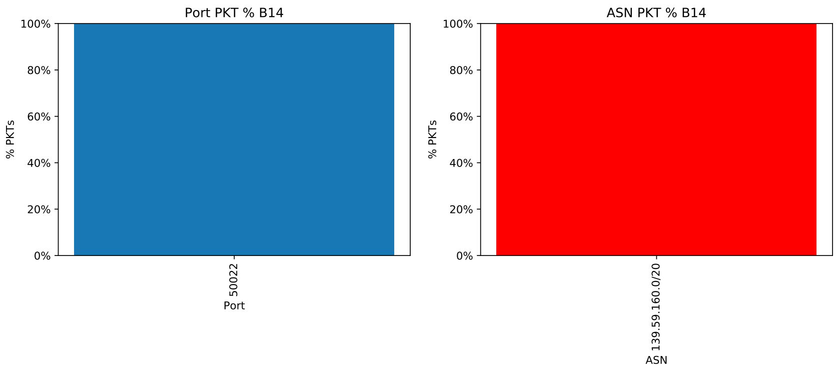


Figure A.10: $\boldsymbol{B_{14}}$

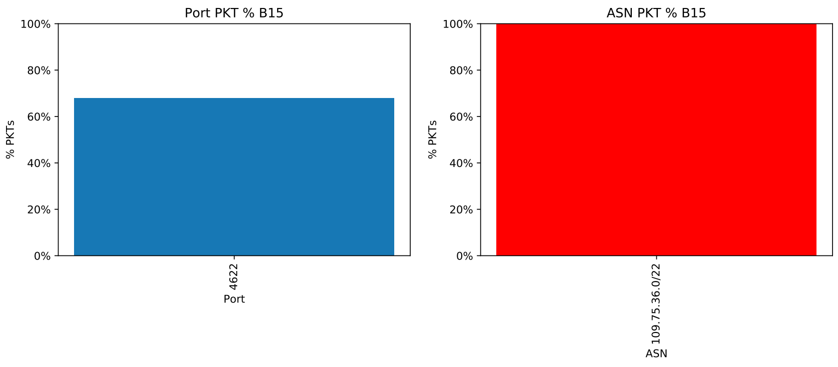


Figure A.11: $\boldsymbol{B_{15}}$

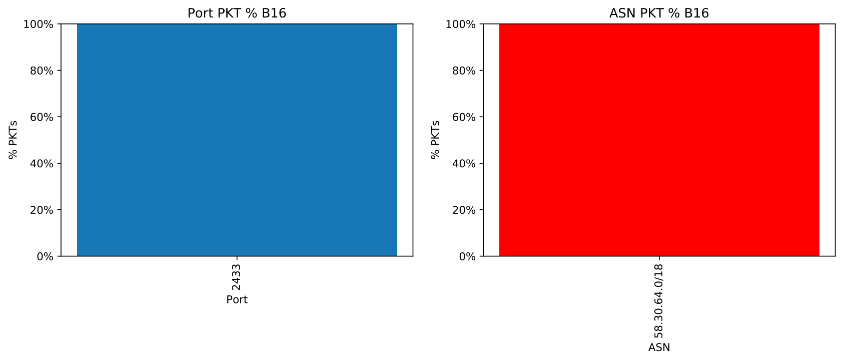


Figure A.12: $\boldsymbol{B_{16}}$

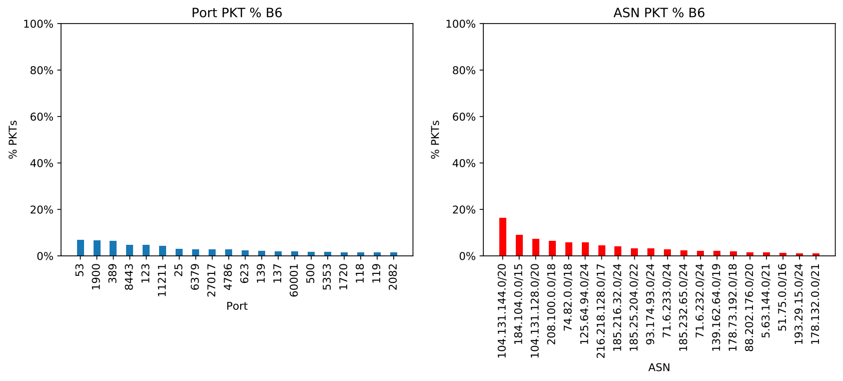


Figure A.13: $\boldsymbol{B_{6}}$

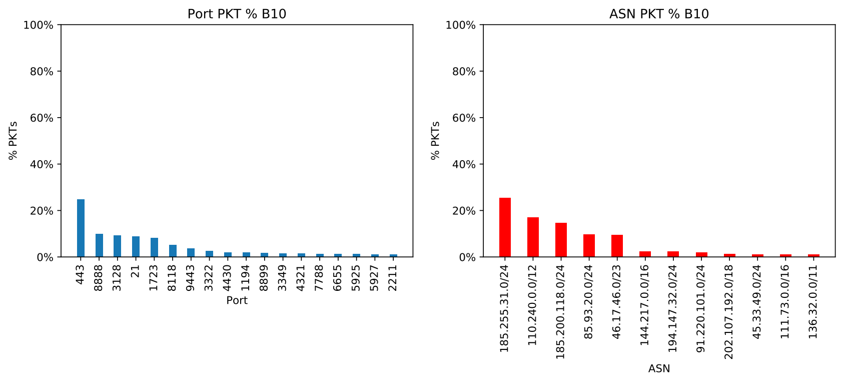


Figure A.14: $\boldsymbol{B_{10}}$

A.2 Communties March 2019

A.2.1 Vertical Scan Mar 2019

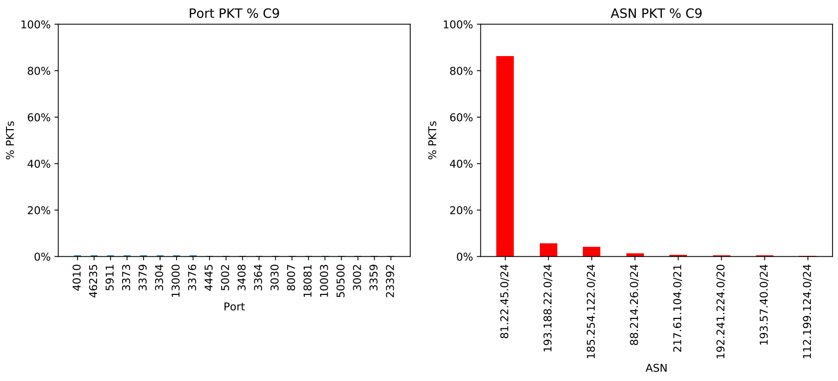


Figure A.15: $\boldsymbol{C_9}$

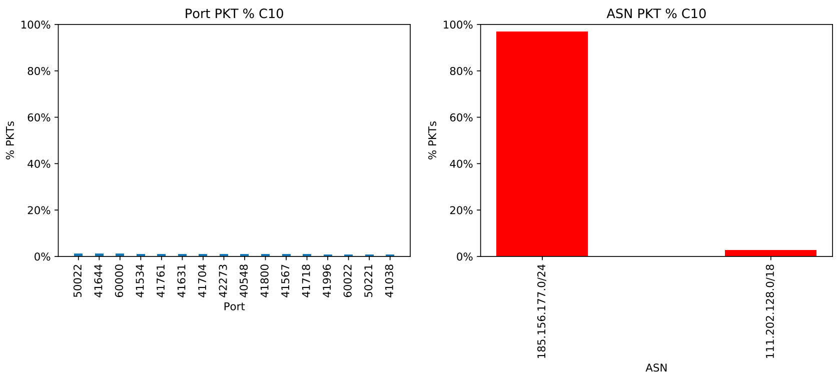


Figure A.16: $\boldsymbol{C_{10}}$

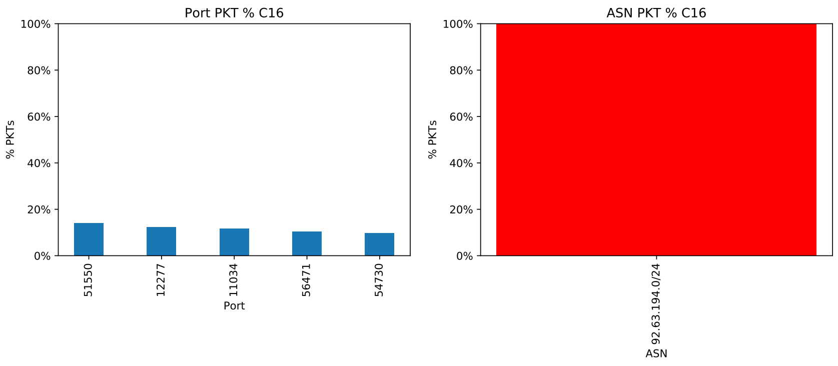


Figure A.17: $\boldsymbol{C_{16}}$

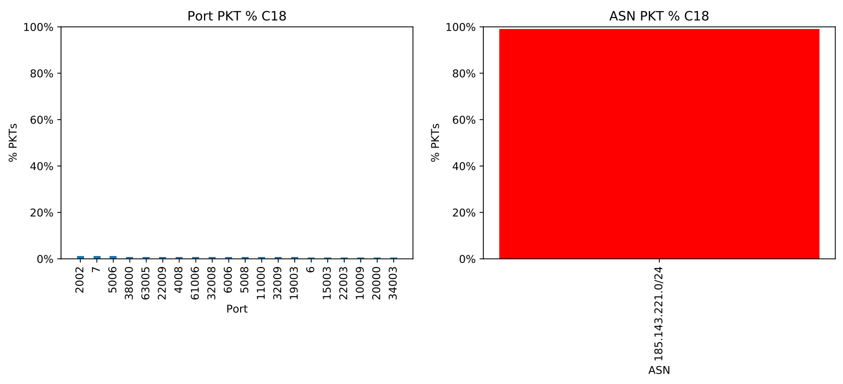


Figure A.18: $\boldsymbol{C_{18}}$

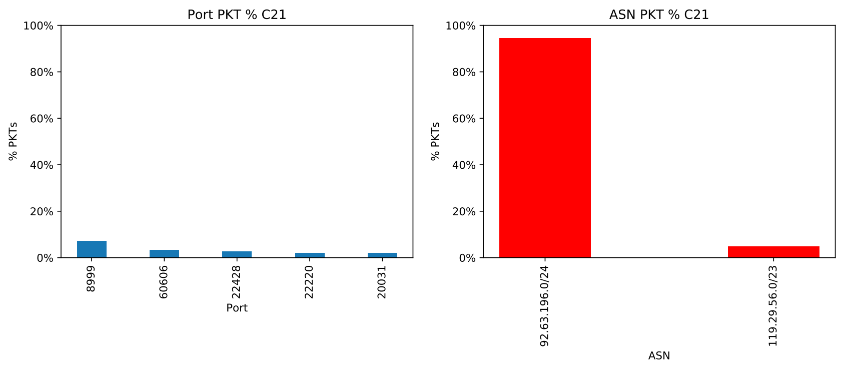


Figure A.19: $\boldsymbol{C_{21}}$

A.2.2 Horizontal Scan Mar 2019

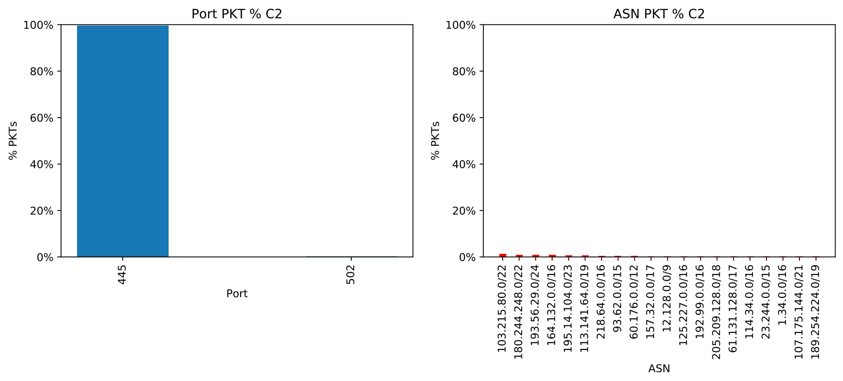


Figure A.20: $\boldsymbol{C_2}$

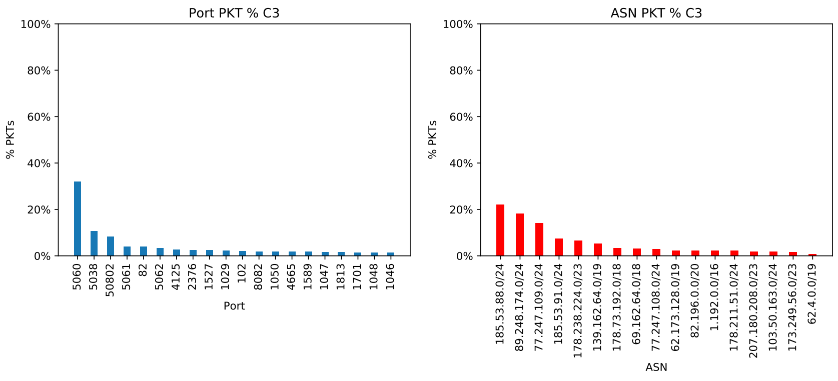


Figure A.21: $\boldsymbol{C_3}$

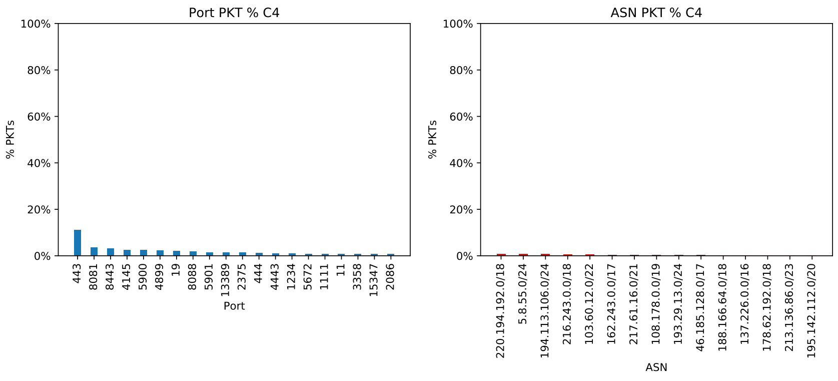


Figure A.22: $\boldsymbol{C_4}$

A.2.3 Miscellaneous Scan Mar 2019

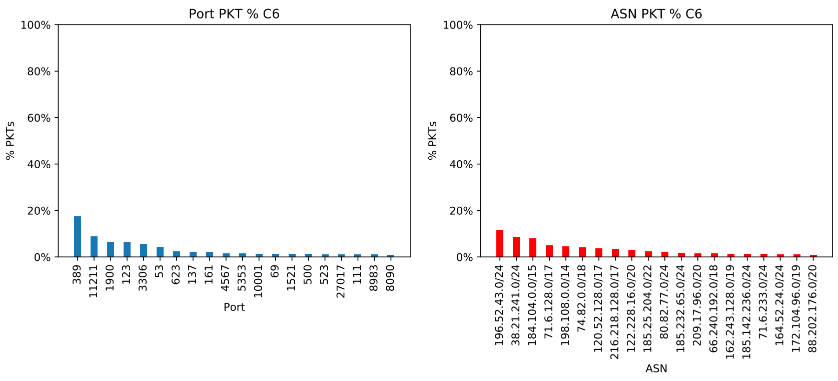


Figure A.23: $\boldsymbol{C_6}$

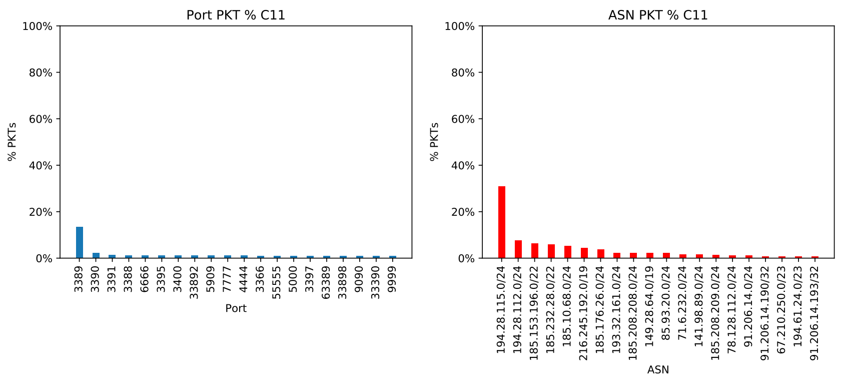


Figure A.24: $\boldsymbol{C_{11}}$

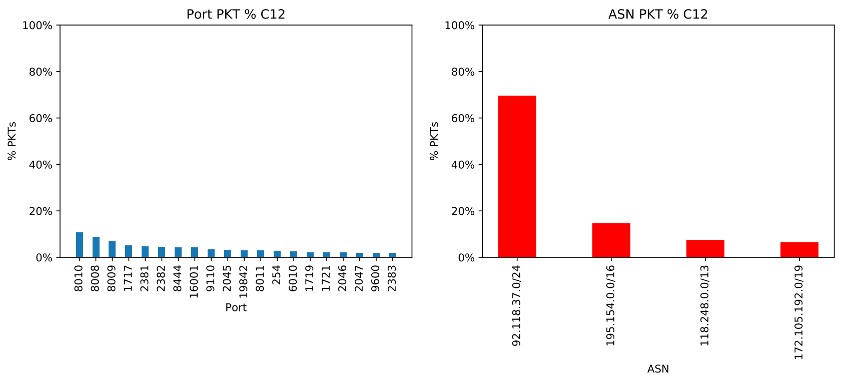


Figure A.25: $\boldsymbol{C_{12}}$

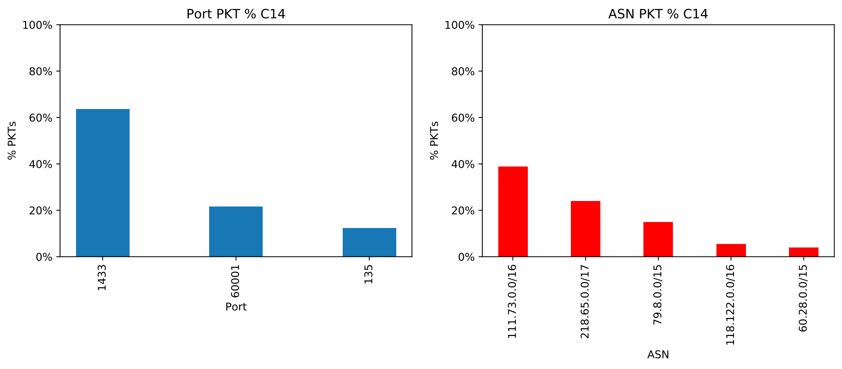


Figure A.26: $\boldsymbol{C_{14}}$

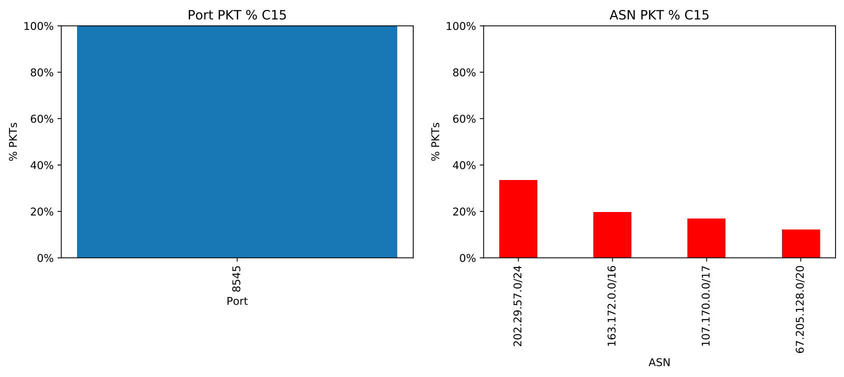


Figure A.27: $\boldsymbol{C_{15}}$

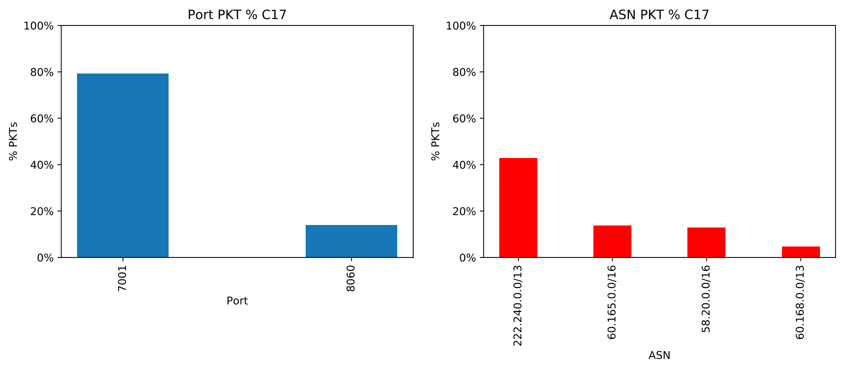


Figure A.28: $\boldsymbol{C_{17}}$

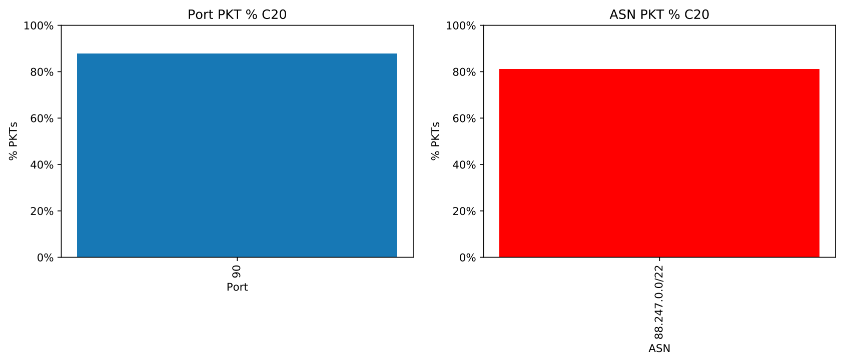


Figure A.29: $\boldsymbol{C_{20}}$

Bibliography

[1] William Aiello, Fan Chung, and Linyuan Lu. A random graph model for power law graphs. *Experimental Mathematics*, 10(1):53–66, 2001.

[2] Michael Bailey, Evan Cooke, Farnam Jahanian, Andrew Myrick, and Sushant Sinha. Practical darknet measurement. In *2006 40th Annual Conference on Information Sciences and Systems*, pages 1496–1501. IEEE, 2006.

[3] Steven M Bellovin and NJ Murray Hill. There be dragons. In *USENIX Summer*, 1992.

[4] Vincent D Blondel, Jean-Loup Guillaume, Renaud Lambiotte, and Etienne Lefebvre. Fast unfolding of communities in large networks. *Journal of statistical mechanics: theory and experiment*, 2008(10):P10008, 2008.

[5] Aaron Clauset, Mark EJ Newman, and Cristopher Moore. Finding community structure in very large networks. *Physical review E*, 70(6):066111, 2004.

[6] Evan Cooke, Michael Bailey, Farnam Jahanian, and Richard Mortier. The dark oracle: Perspective-aware unused and unreachable address discovery. In *NSDI*, volume 6, pages 8–8, 2006.

[7] Evan Cooke, Michael Bailey, Z Morley Mao, David Watson, Farnam Jahanian, and Danny McPherson. Toward

understanding distributed blackhole placement. In *Proceedings of the 2004 ACM workshop on Rapid malcode*, pages 54–64. ACM, 2004.

[8] Zakir Durumeric, Michael Bailey, and J Alex Halderman. An internet-wide view of internet-wide scanning. In *23rd {USENIX} Security Symposium ({USENIX} Security 14)*, pages 65–78, 2014.

[9] Claude Fachkha and Mourad Debbabi. Darknet as a source of cyber intelligence: Survey, taxonomy, and characterization. *IEEE Communications Surveys & Tutorials*, 18(2):1197–1227, 2015.

[10] Santo Fortunato. Community detection in graphs. *Physics reports*, 486(3-5):75–174, 2010.

[11] Michelle Girvan and Mark EJ Newman. Community structure in social and biological networks. *Proceedings of the national academy of sciences*, 99(12):7821–7826, 2002.

[12] Craig Labovitz, Abha Ahuja, and Michael Bailey. *Shining light on dark address space.* Arbor Networks, Incorporated, 2001.

[13] Sofiane Lagraa and Jérome François. Knowledge discovery of port scans from darknet. In *2017 IFIP/IEEE Symposium on Integrated Network and Service Management (IM)*, pages 935–940. IEEE, 2017.

[14] Renaud Lambiotte, J-C Delvenne, and Mauricio Barahona. Laplacian dynamics and multiscale modular structure in networks. *arXiv preprint arXiv:0812.1770*, 2008.

[15] David Moore, Colleen Shannon, Douglas J Brown, Geoffrey M Voelker, and Stefan Savage. Inferring internet denial-of-service activity. *ACM Transactions on Computer Systems (TOCS)*, 24(2):115–139, 2006.

[16] Mark EJ Newman. Fast algorithm for detecting community structure in networks. *Physical review E*, 69(6):066133, 2004.

[17] Usha Nandini Raghavan, Réka Albert, and Soundar Kumara. Near linear time algorithm to detect community structures in large-scale networks. *Physical review E*, 76(3):036106, 2007.