POLITECNICO DI TORINO

Master's Degree in ICT FOR SMART SOCIETIES

Master's Degree Thesis

Federated learning for network traffic analysis

Supervisors

Prof. Marco MELLIA

Prof. Luca VASSIO

Candidate

Kai HUANG

July 2023

Abstract

Darknets are formed by ranges of IP-addresses that do not host services. Darknets constantly receive and record unsolicited traffic, making them valuable instruments to characterize and detect Internet-wide events such as the spreading of new malware, network scans and misconfigurations. Darknets observe scanning activities from thousands of sources, analyzing darknet traffic and detecting coordinated activities can provide meaningful information for network security analysis to detect cyber threats and to counter them more effectively.

Methods like DarkVec, inspired from Natural Language Processing to utilize word embedding Word2Vec for darknet traffic analysis, can extract meaningful insight from large amounts of data to learn representations of activities associated with IP addresses. IP embeddings generated by DarkVec can provide useful insight into coordinated activities but can only provide a limited view since they are built on a single network. To overcome this limitation and obtain a general overview, common representations need to be created from different networks. The huge volume of darknet traffic makes sharing row data from different darknets impractical, so approaches to expand knowledge also should avoid sharing data. Given the lack of comprehensive ground truth available for learning activity patterns in darknet traffic, unsupervised clustering techniques are applied to identify source IP addresses that act in similar patterns. Automatic detection of changes in activities in darknets is crucial to unveiling and mitigating potential cyber threats, making it important to track the evolution of clusters. This work focuses on two aspects: i) leveraging the Federated Learning method to build common representations for source IP addresses observed in different darknets; ii) monitoring evolution of clusters over time to detect temporal changes in darknet.

I develop a federated learning approach for Word2Vec algorithm to learn common representations collaboratively from different darknets. I design a new strategy to aggregate models with different dimensions, make it scalable to multiple networks, perform in-depth analysis in different scenarios to stress the performance. The quality of the common representations is tested and analyzed using real-world data, utilizing domain knowledge of IP addresses belonging to well-known Internet scanners as ground truth classes to provide reasonable validation. Federated learning does improve the quality of representations, for example when two /24 darknets learn collaboratively, weighted F1 score improves from 0.83 in the local training scene to 0.88, the coverage of each participant also increases with about 11,000 more IP addresses.

To unveil the evolution of clusters, I employ and adjust new metrics for tracking transitions in clusters over time to detect changes in the whole clustering rather

than only one cluster. Conventional metrics like silhouette and adjusted Rand Index cannot provide enough insight to detect cluster evolution, but the new method can identify changes like a new cluster emerging reflecting a new coordinated activity, does a cluster consist of existing or the behavior changes, which helps to identify and understand some events and activities.

The results indicate learning representation collaboratively can extract more information and obtain a more general overview of coordinated activities and the method to monitor cluster evolution can provide meaningful insight to detect and analyze changes of those activities.

Table of Contents

List of Tables

List of Figures

Chapter 1

Introduction

1.1 Overview

Network monitoring and measurements play a critical role in cybersecurity. They involve the continuous observation and examination of network traffic to detect anomalies, inspect performance, and ensure compliance with predefined standards. These tools can be used to detect and mitigate potential threats, minimizing the risk of cyber-attacks and privacy leakages, enhancing cybersecurity measures by enabling the identification of vulnerabilities, preventing unauthorized access, and ensuring system resilience. Through the constant evolution of these monitoring and measurement methods, organizations can effectively protect their networks in the face of cyber threats.

Darknets, also known as network telescopes, are network sensors formed by ranges of IP addresses that do not host any services. Darknets constantly receive unsolicited traffic, referred to as Internet Background Radiation (IBR), which can be the result of misconfiguration, backscattering, scanning activities, etc. The Darknet traffic exhibits distinct characteristics that set it apart from conventional internet traffic, it can be used to detect and characterize Internet events, analyzing those unsolicited packets offers several key insights to identify and mitigate cyber threats.

IBR is of considerable volume, incessant, and originates from a variety of services, for instance only a /24 darknet receives millions of packets per day (Chapter 4), manually extracting information from IBR is infeasible, to obtain meaningful insights from darknet traffic, automatic methods should be applied. Solution like DarkVec [1] exhibits decent performance to analysis darknet traffic with Natural Language Processing method.

Most existing studies have focused on single darknets. However, it is important to note that the traffic observed in darknets deployed across different IP ranges can

vary significantly and the sizes of IP ranges allocated for darknets can also vary considerably [2]. Some studies, such as [3], explore data from multiple darknets. However, these studies require the consolidation of data for joint analysis. In practical terms, countering potential threats requires timely information, and sharing data in real time is not possible given the massive volume of darknet traffic data and issues such as data privacy regulations. Therefore, it is necessary to extract information across different networks using methods that do not require sharing of data.

Furthermore, the activities in darknets are ever-changing, such as attacks emerges as new vulnerabilities announced. Changes in the temporal actvities patterns may associated to potential attack, like Mirai botnets change it scanning behavior before widely spread [4]. Clustering are widely used for darknet traffic analysis, so monitoring evolution in cluster over time is a good way to highlight changes in the activity pattern.

The contributions of my thesis are as follows: I develop a federated learning solution for darknet traffic analysis to automate the construction of global representations across different darknets, the method guarantees the quality of the representations while avoiding the sharing of data. I then apply clustering and leverage a new method to track cluster evolution over time. The proposed methods are evaluated on real-world data.

1.2 General pipeline

Extracting meaningful information from the darknet involves a series of operations (see Figure 1.1). The first step is to collect all packets received by the darknet. Next, the raw data are pre-processed, which may include employing filters and create representations, where each sender IP address is mapped to a vector tobe embedded in the latent space. Recently, self-supervised learning methods have become the common approach for accomplishing this stage. By obtaining expressive representations, it becomes possible to apply machine learning algorithms to solve specific downstream tasks. The results derived from these algorithms can provide valuable insights for security analysts, enabling them to understand ongoing activities and identify new threats, among other important aspects.

The primary objective of this thesis is to enhance the initial steps of the darknet traffic analysis process. Rather than relying solely on traffic collected from a single darknet, the approach employed in this study involves leveraging data from multiple networks. To achieve this, federated learning is utilized, enabling the integration of information from various local representation learning models and the construction of common representations (as illustrated in Figure 1.2).

Another significant aspect of this work is to address the downstream task, as

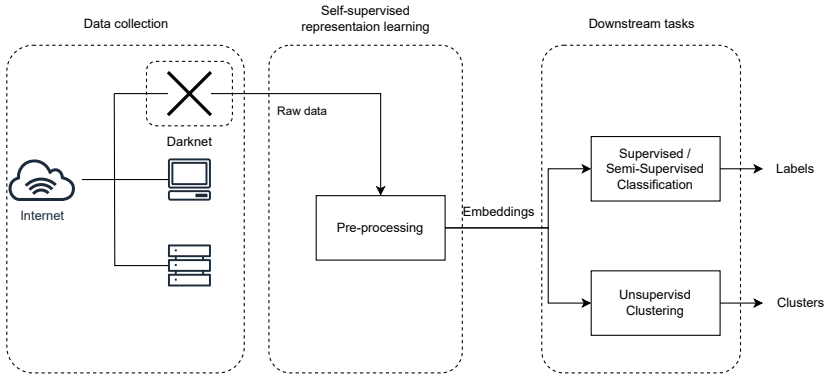


Figure 1.1: Darknet traffic analysis pipeline.

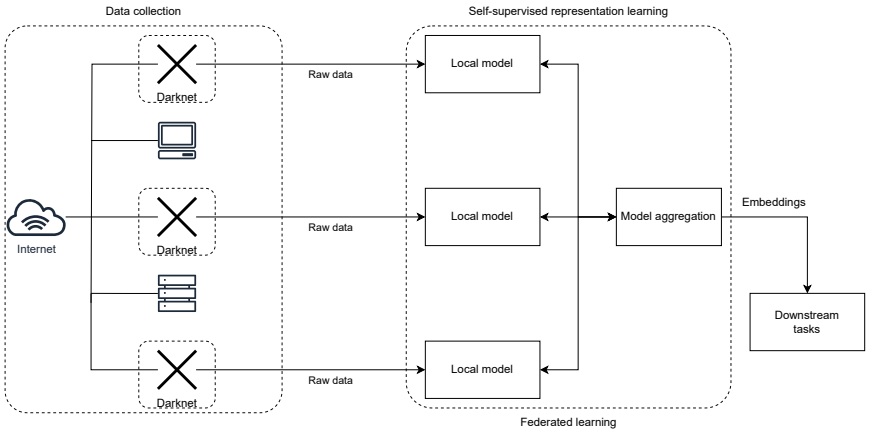


Figure 1.2: Darknet traffic analysis with federated learning.

illustrated in Figure 1.3. Instead of solely performing a clustering algorithm and analyzing the results on a single snapshot, the approach taken in this study aims to leverage the clustering results obtained from different time slots. By continuously monitoring the temporal changes in these results, additional information can be automatically extracted.

1.3 Thesis structure

The rest of this thesis is organized as follows:

- **Chapter 2** provides an overview on the state. In this chapter I briefly review studies on darknet traffic studies and summarize some recent works on

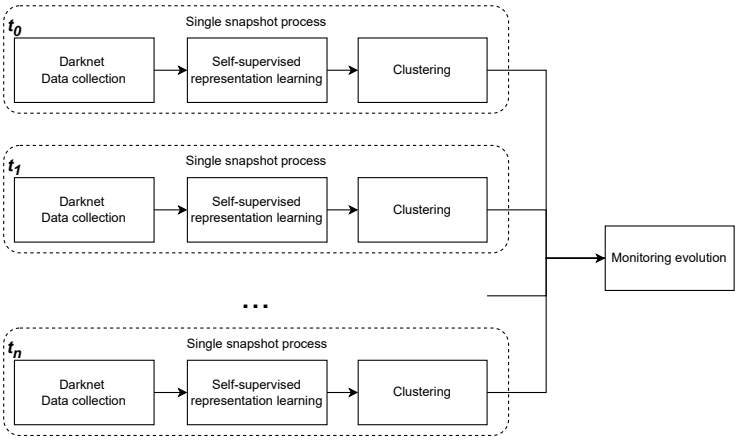


Figure 1.3: Darknet traffic analysis with federated learning.

clustering applied on network traffic. Also a summary of federated learning and its application on cybersecurity is provided in this chapter.

- **Chapter 3** provides the information about the dataset, introduces properties of darknet traffic, gound truth and external information used to characterize darknet traffic.
- **Chapter 4** introduces the methodology for generating embeddings of source IP addresses and for sharing information among different darknets,I explain the utilization of federated learning in this application in detail.
- **Chapter 5** demonstrates the evalution of IP embeddings. I assess the method proposed in Chapter 3 in different scenarios and analyze the results.
- **Chapter 6** introduces the algorithm for clustering darknet traffic and the new metrics used to monitor evolution of clusters over time.
- **Chapter 7** illustrates the result of clustering and the output of the clustering changes observed using different metrics. Then I decribe some discoveries by tracking cluster evolution with new metrics.
- **Chapter 8** draws the coclusion and proposes future improvement of this work.

Chapter 2

Related work

2.1 Internet Measurement Studies

Darknet is the common tool to perform passive network measurement. In recent years, there has been an increasing amount of literature on darknet and analyzing darknet traffic. As early as 2004, some studies [5], [6], [7] have been dedicated to investigating traffic destined for unused IP space. They have exploited this traffic to gain insights into network phenomena like worm propagation, DDoS attacks, and have proposed efficient measurement methods for such traffic.

With the significant changes in the type of malicious activity on the Internet, as well as the evolving quantity and quality of unused address space, the study of darknet traffic has continued and play an important rule in detecting and characterizing malware, for instance, in [4], the authors leveraged darknet data to identify and track the spread of the notorious Mirai botnet in late 2016 and get insight of the botnet's behavior.

Darknet monitoring has the potential to go beyond detecting attacks and scanning activities, it can also provide valuable insights into network infrastructure. In [8], the authors utilize darknet traffic to infer the utilization of IPv4 addresses.And in [9], the study investigates the analysis of macroscopic Internet events, such as network outages, using darknet traffic.

With the popularity and enhancement of machine learning and data science techniques, more recent attention has been focused on the application of these techniques to the darknet traffic analysis. In [10], authors extract spatiotemporal features from packets by considering the source host and destination port within a specified sampling time interval and utilize these features in anomaly detection algorithms to classify malware activities.

Another relevant example is presented in [11], where the authors consider the

headers of TCP packets collected from darknet traffic. They employ a genetic algorithm to aid in the identification of previously unidentified fingerprints associated with scanners. And this approach contributes to the detection of new malware variants.

Another study [12] focuses on the classification and inference of impaired IoT devices at an Internet-scale. The study introduces a novel darknet sanitization probabilistic model to identify and filter out misconfiguration traffic flow, which leverages the targeted destinations associated with the corresponding source. Moreover, the study employs CNN and RF classifiers to identify whether a malicious source is a compromised IoT device, these classifiers utilize various features such as packet length and TTL.

There are studies closely related to the approach used in this thesis that employ natural language processing (NLP) to automate traffic analysis, create numerical representations for senders. Such as DANTE [13] and DarkVec [1], [14]. DANTE uses destination ports sequences from a specific source IP address to create sentences to train Word2Vec embeddings for ports, It treats the port sequence for each sender-receiver pair as a sentence. After obtaining port embeddings, for each sender-receiver pair the representation of source IP address is computed by averaging its port embeddings. The method need train an independent model for each sender, so it has limitation to handle large dataset which records tens of thousands distinct senders.

Works in this thesis are based on DarkVec [1], [14], which generates corpus from source IP addresses sequences based on targeted services (destination ports). DarkVec only requires training single Word2Vec model for all senders, which means it overcomes the scalability limitation of DANTE. A More detailed review of this approach is provided in Section 3.1.

2.2 Federated learning

In recent years, federated learning has gained significant attention as a distributed learning technique that enhances privacy preservation. Its objective is to address the challenge of data silos, where data exists in isolated islands. Traditional machine learning predominantly employs a centralized method to train models, necessitating transferring of collected data from different sources to the same server. However, this becomes challenging when handling substantial volumes of user-generated data and due to mounting privacy concerns and regulatory requirements. Federated learning allows individual users to collaborate and learn machine learning models without sharing their data.

The concept of federated learning was initially introduced by Google in 2016 [15], where the authors proposed a framework that distributes the training data across

various datasets and learns a shared model by aggregating locally-computed updates. The approach they propose, named *FedAvg*, also aims to optimize communication costs during training. And a lot of following researches working on solve main challenges of federated learning, like the communication efficiency [16], privacy protection [17], [18] and statistical heterogeneity [19]. And [20] categorize federated learning to 3 main classes. horizontal federated learning in which datasets share the same feature space but with different samples, vertical federated learning in which datasets share the same sample ID space but differ in feature space and federated transfer learning in which datasets differ not only in samples but also in feature space.

Majority of federated learning tasks focus on supervised scienarios like classification, but there are still an increasing amount of studies to apply federated learning on unsupervised learning scienarios, like [21] present a framework to collaboratively train an unsupervised deep convolutional autoencoder on healthy magnetic resonance scans data. And the closet study to this thesis work is *FederatedWord2Vec* [22], the authors train Word2Vec embeddings in federated way by extract only common words in different corpus and follow the architecture of FedAvg.

Federated learning is applied in different fields like healthcare [23], [24], and it also applied on cybersecurity like for intrusion detection [25]. The authors proposed an intrusion detection model based on gated recurrent unit and SVM to classify time series data of network traffic. They applied attention mechanism in the federated learning process to adjust weight of important participants, and also optimized transmission cost by reducing the upload of unimportant updates.

In [26] authors proposed a risk intelligence system in which different service provider collect data from users and construct a global risk model with federated learning. The study in [27] presents a framework to build a global model for identify malicious HTTP communications, it utilizes federated learning to train FastText embeddings to get representation for URL, and also leverage federated learning in downstream tasks to classify malicious activities.

2.3 Unsupervised learning

As described in Section 1.2, once representations for darknet traffic are computed, the last step of the processing pipeline can involve either a supervised or unsupervised analysis. Given the fact that there is a lack of ground truth in darknet traffic, the unsupervised learning becomes a more powerful approach.

Clustering is a unsupervised machine learning technique that groups similar data samples together based on their patterns and characteristics. It is commonly employed in network traffic analysis, particularly in darknet scenarios. Analyzing darknet data with clustering enables the identification of novel attack patterns,

attacked victims, and new network scanners.

In [28] the authors uses information carried in the packets, such as port, protocol, TTL (time-to-live), construct vector of 22 features for source IP address add by taking their statistical mode, and clustering traffic from darknet to identify activities like long-term scanning and bursty events from attacks.

In [29], the authors extract features such as traffic volume and scan strategy for source IP addresses. They construct representations using an autoencoder and employ K-Means clustering for analyzing a significant volume of darknet data. Furthermore, the studies model temporal changes in darknets as an optimal mass transport problem and utilize the optimal distance to quantify changes in the overall darknet landscape.

Some studies also use graph based approaches to cluestering darknet traffic, [30] builds a bipartite graph from source IP addresses and destination ports, and use Louvain algorithm [31] to detect communities characterized by similar target ports. And DarkVec[1] construct nearest neighbor graph based on embeddings and use Louvain method to clustering source IP addresses.

Regarding the clustering of other types of network traffic, [32] present an unsupervised approach for detecting changes in the YouTube CDN cache selection policy. They employ HDBSCAN to cluster caches and monitor the evolution by measuring the movement of the cluster centroid. LENTA [33] aims to cluster URLs. The study proposes an iterative approach that employs DBSCAN with different ϵ values to extract well-shaped clusters. Additionally, the study builds system knowledge about the obtained clusters by sampling clusters and labels newly discovered clusters by comparing them with previous clusters.

Chapter 3

Background

3.1 i-DarkVec

i-DarkVec is a method utilizes Natural Language Processing techniques to analyze darknet traffic, it leverages Word2Vec embeddings to capture meaningful representations of source IP addresses reached darknets, referred to as IP embeddings. The main idea is that senders perform similar activities reach darknet at nearby time and targeting the same service, which is similar to words co-occur in sentences. According to result of [14], this method can generate robust representations to extract complex activity patterns from raw darknet traffic.

3.1.1 Word2Vec

The technique used by i-DarkVec is Word2Vec, which is a neural network model to learn representations for words from a large corpus of text. Word2Vec can use two different model architectures: Continous Bag Of Words (CBOW) and skip-gram. The latter is utilized in i-DarkVec, Figure 3.1 shows the model with skip-gram architecture, which aims to use the input word to predict the surrounding context words. The input is an one-hot vector, which dimension is equal to the vocabulary size $\boldsymbol{V}$ (number of distinct words in the corpus). A $\boldsymbol{V} \times \boldsymbol{N}$ matrix $\boldsymbol{E}$ represents the weights between input and hidden layer, each row of this matrix is the $\boldsymbol{N}$ dimension representation of the corresponding word, i.e. embedding. The output layer outputs $\boldsymbol{C}$ multinomial distributions using the same $\boldsymbol{N} \times \boldsymbol{V}$ weight matrix $\boldsymbol{E'}$, where each column corresponds to a word in the vocabulary, in the python library gensim[34], the transpose of this matrix is represented as a vector *syn1neg*, in the following section I use $\boldsymbol{S}$ to denote this vector.

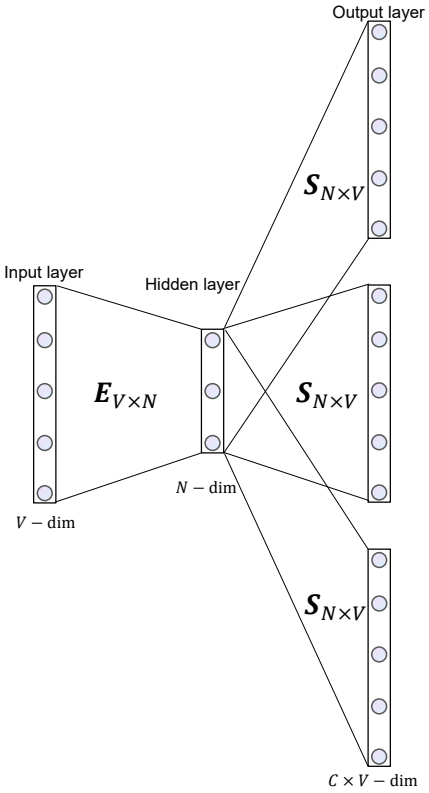


Figure 3.1: The skip-gram model

3.1.2 Corpus generation

Before training the Word2Vec model, i-DarkVec requires a pre-processing stage to creates corpus from raw packets. The first step is filtering low-traffic senders, IP adderesses that sending more than a specific number of packets m_{pkt} to the darknet within the given time period are considered as *active* senders. i-DarkVec selects only active senders, the senders present in the time period but not active will be discarded.

After filtering, the following step is to construct "documents". Each active IP address is considered as a word, then documents are created as the sequence of IP addresses in Chronological order. To highlight similarities among senders' activity, i-DarkVec separates senders into different groups by *services*. Services are defined by *destination ports*. i-DarkVec achieves best performance with *Auto-defined services*, which means each of the top-n popular ports corresponding to a specific service, and all the remaining ports form another service. As the example shown in Figure 3.2, in each time period i-DarkVec create one sequence for each service, thus there are $n + 1$ sequences in the corpus for one time period.

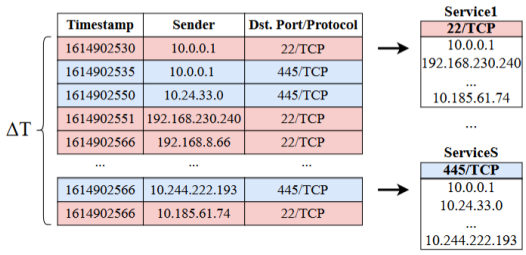


Figure 3.2: Corpus generation

3.1.3 Incremental training

Unlike NLP problem that to train embeddings for words from a large corpus in a single round, senders are observered incrementally when they reach darknets, thus i-DarkVec continuously create and update embeddings for IP addresses, as shown in Figure 3.3 it extract corpus from traffic observed within a time window, train a Word2Vec model, and as time passes, when new batches of traffic is collected, it updates the previous model with new corpus and get new embeddings.

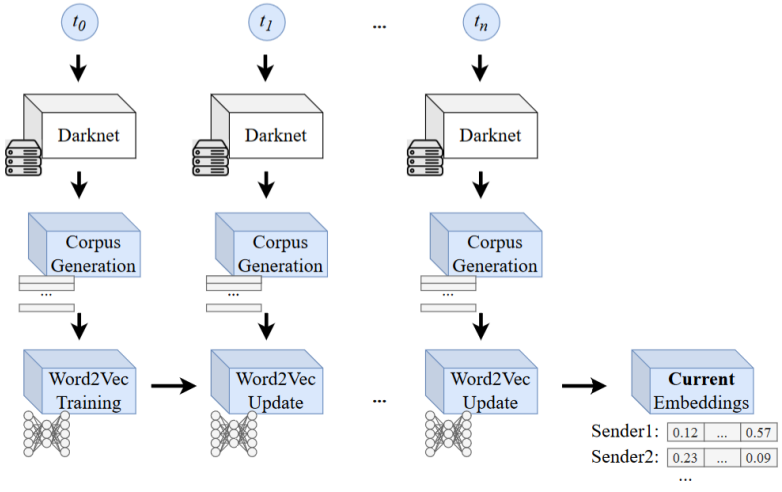


Figure 3.3: Incremental training in i-DarkVec

3.2 Clustering algorithm

Given the fact that majority of observed IP addresses are unlabelled, it is necessary to leverage unsupervised learning. The result in 6 shows senders from same source are close to each other in the latent space, prove the feasibility of using clustering method to identify coordinated activities.

Considering darknet traffic is very noise, activity patterns of some senders are irregular, the clustering need to be robust to outliers. And given the significant complexity of parameter tuning, the algorithm should ideally be parameter-free, i.e. result is not significantly affected by hyper-parameters. So I use HDBSCAN[35][36] to clustering IP addresses, HDBSCAN is a density-based clustering method, while unlike DBSCAN[37], which relies on the hyper-parameter ϵ to determine the density of clusters, HDBSCAN has the ability to automatically determine the appropriate density threshold for defining clusters. Moreover, HDBSCAN effectively identifies clusters with varying densities, accommodating both clusters with diverse internal densities and clusters with differing densities across different regions of the dataset. Additionally, it extracts only the most significant clusters, enhancing its robustness against noise.

The only mandatory parameter of HDBSCAN is m_{pts}, it defines the minimum size of a cluster and it also works as a smoothing factor in density estimation. The procedure of HDBSCAN can be divided into the following steps:

- **1. Transform the space** For each sample x in a dataset $\mathbf{X}$, its *core distance* $d_{core}(x)$ is defined as the distance to the m_{pts}-th nearest neighbor. To spread apart points with low density, HDBSCAN defines a new distance metric between points called *mutual reachability distance* as follows:

$$d_{mreach}(a, b) = \max\{d_{core}(a), d_{core}(b), d(a, b)\} \tag{3.1}$$

  where $d(a, b)$ is the original distance between the two samples. After the first step the original distance matrix is transformed into a mutual reachability distance matrix.

- **2. Build the Minimum Spanning Tree (MST)** A MST is a subset of edges that connects all vertices of a connected undirected graph, without any cycles and with the minimum total weights of edges. In this step, HDBSCAN constructs a MST for the graph generated from the dataset, where each vertex represents a sample, and the weight of an edge between two samples corresponds to their mutual reachability distance.

- **3. Build the cluster hierarchy** In this step, the MST is converted into the hierarchy of connected components. All edges from the MST are removed iteratively by decreasing order of weights (distances) to obtain a dendrogram.

- **4. Condense the cluster tree** Before extracting significant clusters, the dendrogram needs to be condensed. Instead of perceiving a cluster split as the creation of new clusters, it is viewed as a persistent cluster losing points. By traversing the hierarchy and evaluating splits, clusters with fewer points than m_{pts} are considered "falling out" and are merged with the larger cluster. Splitting into two clusters of m_{pts} or larger is recognized as a true split. The resulting tree provides information on cluster size reduction as distance varies.

- **4. Extract clusters** To achieve a flat clustering that extracts only significant clusters, it is necessary to select only the clusters with higher persistence. A metric $\lambda = \frac{1}{distance}$ is used, for a cluster λ_{birth} and λ_{death} represent λ values when it has been split from other clusters and split into two separate clusters. And λ_p denotes the value when a point p in that cluster leaving it, either real split or "falling out". The *stability* of each cluster is computed as

$$\sum_{p \in cluster} (\lambda_p - \lambda_{birth}) \tag{3.2}$$

  When travelling up through the tree, if the sum of child cluster stabilities is higher than the cluster's stability, the cluster's stability is updated as the sum. Otherwise, if the cluster's stability is higher, it will be selected. The selected clusters form the final flat clustering.

Chapter 4

Dataset

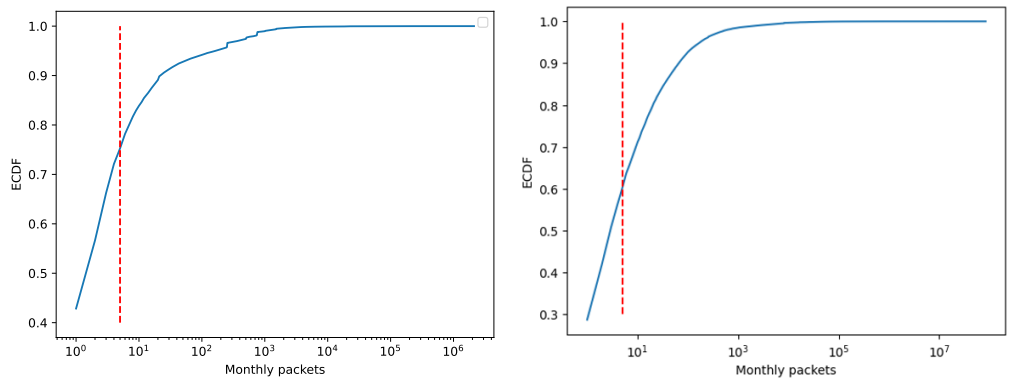
4.1 Overview

My work relies on traffic captured from two darknets, one is a /24 IPv4 network hosted by Polito an the other one is a /19 network in Brazil. I use the traces collected in May and June of 2021, Table 4.1 provides some basic statistics for these two darknets. The Brazilian darknet has 32 time the IP range of Polito Darknet, and it collected more than twenty times its traffic. The larger darknet can observe more activities, it can collect traffic targets for all ports in a single day. In contrast, the large darknet does not observe dozens of times more IP addresses, most of senders recorded by Polito darknet are also oberserved in the other one, but there are still about 53K source IP addresses only appeared in Polito darknet.

	Polito /24 Darknet			**Brazilian /19 Darknet**		
Period	**Packets**	**Sources**	**Ports**	**Packets**	**Sources**	**Ports**
31 days	64 189 909	532 609	65 537	1 541 048 201	3 021 317	65 536
Last day	2 573 967	44 850	30 010	56 113 778	287 001	65 536

Table 4.1: Basic statistics for two darknets in May 2021.

Figure 4.1 shows the Empirical Cumulative Distribution Function (ECDF) of the number of packets sent by each source in one month. most source IP addresses observed in darknet sent very few packets, in Polito darknet 42.8% of them are captured just once in the whole month and 75.3% of them sent no more than 5 packets. To limit "noise" in the data, it is necessary to filter senders that transmitted too few packets, those sources are likely to be randomly-spoofed addresses or misconfigurations. While since Brazilian darknet has wider IP range, it is easier to log more traffic from those random senders the fraction of senders with very few packets is smaller.



(a) Polito /24 darknet **(b)** Brazilian /19 darknet

Figure 4.1: Amount of packets of each sender.

The dataset contains various raw features for traffic characterization. Protocols and destination ports are frequently employed to identify targeted services. Table 4.2 presents several protocol types, while traffic with other protocols is negligible. Packets with TCP protocol constitutes the majority of the traffic; however, there is a noteworthy surge in GRE (Generic Routing Encapsulation) protocol traffic observed on the final day. In fact, more than half of the GRE traffic collected throughout the month is concentrated on the last day. This thesis also successfully identifies the network events associated with this traffic, with further elaboration provided in Section 8.3.2. As for Brazilian darknet (see Table 4.3), the overall situation is similar but it observed more UDP traffic. The fraction senders using UDP protocol in Brazilian darknet is significantly higher than Polito darknet. It also recorded a clear upward trend in GRE traffic at the end of the month. The growth in traffic is even more pronounced, but the growth in the number of senders is not significant compared to the total number of IP addresses it observes.

Protocol	**31 days**		**Last day**	
	Packets(%)	**Senders**(%)	**Packets**(%)	**Senders**(%)
TCP	94.07	77.95	94.94	82.92
UDP	5.53	21.40	4.16	13.35
ICMP	0.35	2.04	0.28	1.62
GRE	0.05	0.70	0.62	4.78

Table 4.2: Protocol types of Polito /24 darknet traffic in May 2021.

Tables 4.4 shows the top 3 targeted ports in Polito darknet in that month, out of the 2^{16} ports, ports 23, 22, and 5555 received the highest amount of traffic. Port 5555 is commonly associated with Android Debug Bridge (ADB), while port 23

Protocol	31 days		Last day	
	Packets(%)	**Senders**(%)	**Packets**(%)	**Senders**(%)
TCP	90.20	59.65	92.03	67.60
UDP	9.23	38.00	6.60	29.06
ICMP	0.51	2.11	0.44	1.92
GRE	0.06	0.23	0.92	1.42

Table 4.3: Protocol types of Brazilian /19 darknet traffic in May 2021.

is frequently targeted by the Mirai botnet [4]. While the landscape in Brazilian darknet is quite different, in this darknet port 5555 is not among the most targeted ones, and there is a larger volume of traffic targets port 445, which is used for Microsoft Server Message Block (SMB), also has long been abused for malicious activities.

Period	Port	Traffic(%)	Sources
31 days	5555	5.50	21 740
	22	4.58	15 776
	23	2.64	189 165
Last day	5555	6.44	1621
	22	4.74	1795
	23	2.07	15403

Table 4.4: Top 3 ports of Polito /24 darknet traffic in May 2021.

The statistics from various darknets reveal significant variations in activities across different darknets. Even in very large darknets, there are limitations to the visibility of overall activities. The restricted view obtained from a single darknet reduces the ability to acquire comprehensive knowledge about darknet activities. To overcome this limitation, it is preferable to leverage observations from multiple darknets.

Similarly, the traffic observed by a darknet can change dramatically from one day to the next without prior indication. Detecting and interpreting the temporal change may provide meaningful information.

4.2 Ground Truth

One of the biggest challenges of darknet traffic analysis is the lack of ground truth, and I used domain knowledge to label the IP addresses of the sources. Some of the labels come from prior knowledege about widely known scanning projects. Others

Period	Port	Traffic(%)	Sources
31 days	23	4.12	675431
	22	3.33	107 807
	445	1.76	582 311
Last day	23	3.89	76 874
	22	2.96	8 685
	445	1.99	5 7359

Table 4.5: Top 3 ports of Brazilian /19 darknet traffic in May 2021.

are constructed based on known fingerprint like Mirai[4]. However majority of observed senders do not have a corresponding label, these senders may be from other sources or some of the known classes, since their sources cannot be identified, I mark them as *Unknown*.

Source	Senders	Packets	Ports	Top 3 ports/protocols (%traffic)
Ipip[38]	36	194 261	31	5060/TCP(69.70), ICMP(3.68), 3128/TCP(3.06)
Netsystems[39]	50	256 862	205	80/TCP(0.96), 943/TCP(0.95), 443/TCP(0.93)
Censys[40]	355	3 678 694	64985	5060/TCP(3.73), 2000/TCP(2.87), 3128/TCP(3.84)
Shodan[41]	36	367 884	1268	2000/TCP(4.18), 443/TCP(1.22), 80/TCP(1.22)
Internetcensus[42]	251	260 413	245	443/TCP(9.28), 2000/TCP(7.55), 5060/TCP(6.88)
Shadowserver[43]	289	383 446	58	123/UDP(3.99), 17/UDP(2.03), 3283/UDP(2.03)
Mirai-like	19 959	1 139 482	1154	23/TCP(90.37), 2323/TCP(3.23), 5555/TCP(1.89)
Rapid7[44]	344	86657	158	443/TCP(2.26), 50880/TCP(1.68), 8181/TCP(1.68)
Umich[45]	50	5004	1474	53/UDP(45.38), 7/TCP(24.74), 48677/TCP(0.06)
Onyphe[46]	130	33003	148	2404/TCP([2.87), 5985/TCP(0.85), 8200/TCP(0.77)
Binaryedge[47]	163	55318	236	5060/TCP(27.20), 443/TCP(1.41), 5901/TCP(1.05)
CAIDA Ark[48]	15	400	-	ICMP(100)
Stretchoid[49]	16	23682	69	2000/TCP(12.74), 9200/TCP(4.27), 44818/TCP(4.25)
Pnap	30	1661	-	ICMP(100)
Securitytrails[50]	18	61714	180	80/TCP(1.24), 443/TCP(1.22), 2121/TCP(1.22)
Cybercasa[51]	253	6975	786	15627/TCP(0.53), 21242/TCP(0.40), 1099/TCP(0.39)

Table 4.6: Activities of ground truth classes in the 31-day dataset.

Table 4.6 shows characteristics of different ground truth classes recorded in the 31-day long dataset of Polito darknet. The classes exhibit substantial imbalance and manifest heterogeneity. Mirai-like botnets consist of thousands of senders, with over 90% of their traffic targeting port 23. In contrast, Censys, a security platform, contains only hundreds of IP addresses and most of them hosted in two /24 subnets, yet it sent millions of packets regularly targeting almost all of the 2^{16} ports. There are certain classes that consist of only very few senders, such as CAIDA's active measurement infrastructure Ark. In a that month, only 15 senders were observed within this class. These senders exclusively transmit ICMP packets; however, they are distributed across different ranges within the IP space.

In general, senders belonging to one class are expected to differ from those belonging to other classes, while differences in the characteristics of different classes may lead to differences in the difficulty of correctly classifying them. Intuitively, classes that have a larger number of samples and exhibit frequent, regular activity

are generally easier to recognize. Conversely, classes with a smaller number of samples, minimal traffic, or more random activity patterns tend to be more challenging to correctly identify.

4.3 External information

Apart from the data recorded in the raw dataset of darknet traffic, to extract further insights about the source or cyber events, additional features are required to characterize these source IP addresses, such as geographic location and domain name. This additional information is obtained from external sources, and it is commonly used when manual inspection is necessary following automated analysis.

4.3.1 Geographic location

MaxMind's GeoLite2 databases [52] are the source of external information regarding geographic location. These databases provide geolocation details for IP addresses, including city, country, continent, as well as information about their corresponding autonomous systems (AS).

Table 4.7 displays the geographic distribution of recorded traffic and observed senders. Darknet activity varies significantly among senders from different regions of the world. Approximately half of the collected darknet traffic originates from Russia, while the number of senders from Russia is only approximately 22,000. Conversely, although there are a large number of IP addresses from China and India that interact with the darknet and send traffic, most of them exhibit low activity levels.

	Packets	**Senders**
Country	Russia(43.52%)	China(21.36%)
	United States(17.54%)	India(9.66%)
	China(8.75%)	United States(6.63%)
	Netherlands(2.98%)	Brazil(6.27%)
	France(2.89%)	Russia(4.26%)
Continent	Europe(58.75%)	Asia(55.08%)
	North America(19.60%)	Europe(21.46%)
	Asia(19.02%)	North America(10.37%)
	South America(1.63%)	South America(9.53%)
	Africa(0.42%)	Africa(2.81%)

Table 4.7: Top 5 countries and continents of Polito /24 darknet in May 2021.

Considering that the Polito darknet is deployed in Europe, it may capture

a higher proportion of traffic from Europe, resulting in a potential bias in the observed landscape. Sharing information with networks deployed in other regions of the world could help achieve a more comprehensive global visibility. In fact, if take a /24 subnet from Brazilian darknet (see Table 4.8), it's clear that it collects more traffic from South America and North America, Russia is still the most active one but with smaller percentage.

	Packets	**Senders**
Country	Russia(33.68%)	China(20.23%)
	United States(21.33%)	Brazil(8.97%)
	China(9.54%)	India(8.90%)
	Netherlands(8.47%)	United States(6.09%)
	Canada(3.69%)	Russia(4.06%)
Continent	Europe(51.46%)	Asia(52.34%)
	North America(26.27%)	Europe(19.64%)
	Asia(18.50%)	South America(13.33%)
	South America(2.56%)	North America(11.36%)
	Africa(0.54%)	Africa(2.58%)

Table 4.8: Top 5 countries and continents of a Brazilian /24 darknet in May 2021.

4.3.2 Domain name

Domain name is a important identification of the ownership or control of internet resources like computers and networks. A domain name is a string formed under the rules of the Domain Name System (DNS), DNS can translate domain names into numerical IP addresses, so when the IP address is known, its domain name can be retrieved by reverse DNS lookup query. To get domain names corresponding to sender IP addresses, I used a python module *socket*[1] to perform reverse DNS lookup queries. Socket is an low-level network interface provided by python, and it enables the reverse DNS lookup and other networking functionalities like client-server applications and network protocols.

A full domain name can consist of multiple components, which are concatenated and delimited by dots. The domain hierarchy descends from right to left in the name. Typically, a full domain name can be divided into three main parts. An example is illustrated in Figure 4.2. The term "public suffix" is also known as the effective top-level domain (eTLD). It can be a generic top-level domain (gTLD) (e.g., ".com", ".org"), or it may consist of both a country code top-level domain

[1]https://docs.python.org/3/library/socket.html

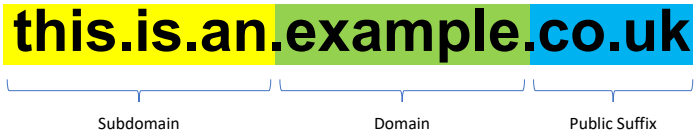


Figure 4.2: An example of a domain name

(ccTLD) and a country code second-level domain (ccSLD), such as "co.uk" and "ac.uk". The middle part refers to the organization that registered the domain name with a domain name registrar. Commonly, it is a second-level domain, but in some cases only third-level registrations are allowed (e.g., co.uk). This part is crucial for identifying the source of an IP address, and for simplicity, it is referred to as the *domain*. A subdomain is a part of the domain. Some organizations, like Google, offer different services with different subdomains, such as "drive.google.com" and "maps.google.com". Both subdomain and public suffix may consist of more than one component. Therefore, it is necessary to use a tool called tldextract[53], which is based on the Public Suffix List[54], to separate these different parts.

Domain name may provide the most straightforward information about the source of an IP address, however it cannot be used for all IP addresses. Consider tens of thousands IP addresses are observed by darknet, performing such a large number of queries requires very frequent requests to DNS servers, and this kind of action may be identified as an attack. In practice, it is very necessary to cache the results of a query to avoid repeated queries. One way to scale up the acquisition of domain names is to access some of the reverse DNS databases, such as Sonar project by Rapid7 [44].

Not all IP addresses can be matched with valid domain names through reverse DNS lookup. A reverse DNS lookup may fail due to several reasons, sometimes there is no valid pointer (PTR) record associated to this IP address is recorded, actually during the experiments, about 25% of queries failed. In cases where no results are returned, the corresponding domain name for the IP address is designated as unknown.

Chapter 5

Creating IP embeddings in federated learning setup

5.1 Simple methods to collaborate

i-DarkVec works well on a single darknet. But in order to share information among multiple different networks, the information produced by each local model needs to be integrated into a unified and comprehensive model. A naive approach is to put all the raw data together to train a global model, but this is clearly infeasible considering the large volume of darknet traffic data. Firstly, we proposed two simple methods to collaborate and share information among different darknets.

5.1.1 Merging the corpus

The first approach still aims to use union datasets but in a more effective way, the idea of this method is to share the pre-processed data instead of the raw data, In the subsequent sections, I will refer to it simply as the **union**.

Figure 5.1 shows the procedure, in i-DarkVec pre-processing means the corpus generation, each day each participant sends its corpus to a central server, then the server uses the union of all received corpus to train or update the global model.

5.1.2 "Ping-pong" approach

This approach is to share the model rather than the data. It takes advantage of the incremental nature of i-DarkVec to enable information sharing. In addition to the incremental training over time, it also conducts incremental training across participants.

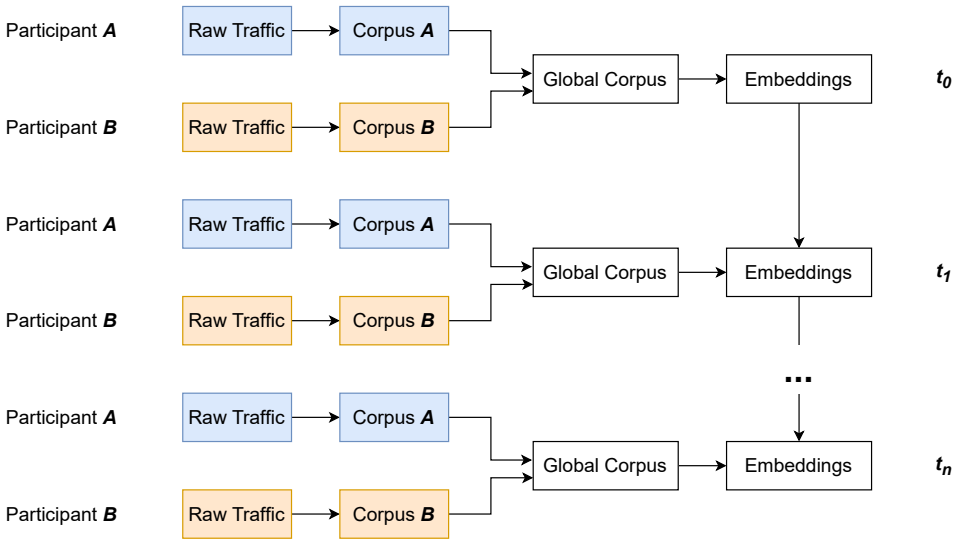


Figure 5.1: Union approach

Figure 5.2 shows an example of two participants, each day participant A creates the corpus from its local data, trains or updates the model and then sends the model to another participant B. After receiving the model, B updates the model with its local corpus in that day and sends the model back to B. Follow this scheme, A and B can create a global model.

The model is repeatedly and continuously transmitted between two participants like they playing ping-pong, so in the subsequent sections, I will refer to it as **ping-pong** approach.

This method has a problem that cannot be ignored, each participant can only start training after receiving the model passed by another participant, as the number of participants increases, the process of passing the model across participant becomes complicated, and waiting for other participants to finish training can be very time consuming.

5.2 Federated learning approach

Federated learning is commonly used to overcome the problem of data silos, it allows multiple users training collaboratively without sharing data. I use a standard federated learning method with a central server on i-DarkVec, architecture is shown in Figure 5.3, information is being shared for every time slots.

In a time step t each user k updates the model in previous day m_k^{t-1} with

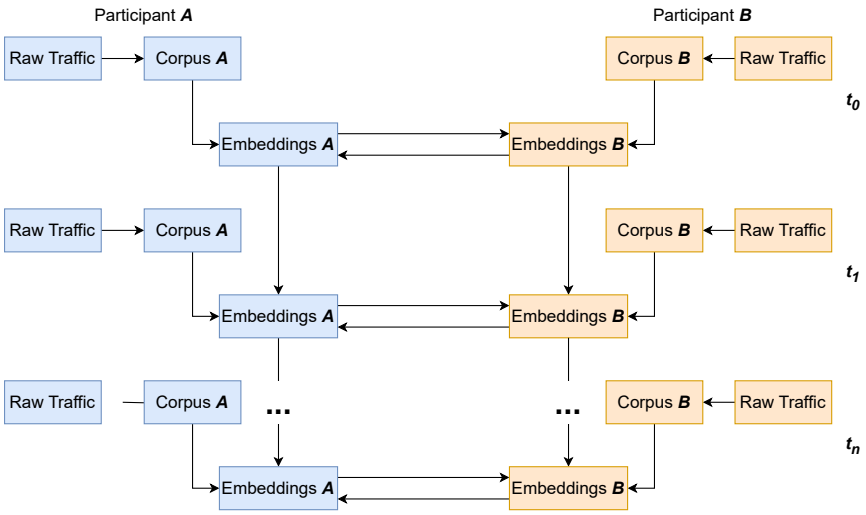


Figure 5.2: "Ping-pong" approach

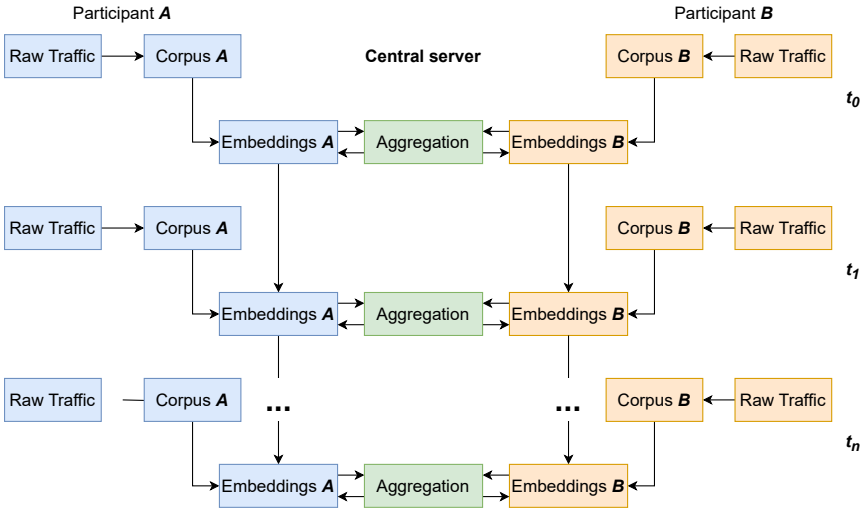


Figure 5.3: Federated learning approach

its local corpus C_k^t and sends current model m_k^t to the central server, the server aggregates received models and sends the unified global m^t model back to every user. This update procedure for a single time window can be performed multiple round. Complete pseudo-code is given in Algorithm 1.

Algorithm 1 Federated learning approach on i-DarkVec at time t. K participants are indexed by k.

for each participants $k \in K$ **do**
    $m_k^t \leftarrow m_k^{t-1}$
end for
for each round **do**
    for each participants $k \in K$ **do**
        $m_k^t \leftarrow LocalUpdate(m_k^t, C_k^t)$
    end for
    $m^t \leftarrow ModelAggregation(\{m_1^t, ...m_K^t\})$
    for each participants $k \in K$ **do**
        $m_k^t \leftarrow m^t$
    end for
end for

5.2.1 Model Aggregation

In the standard federated learning scenario the most common way to perform model aggregation is averaging, but unlike standard scenario which all participants use models with same dimension, in a Word2Vec model the size of input and output layer are depends on the vocabulary and each weight corresponds to a distinct word. In the i-DarkVec scenario, the vocabulary is the active IP addresses and different darknets collect traffic from different sources, which means participants hold models with different sizes and some IP addresses are not be observed by some of the participants, this fact makes it impossible to simply perform model averaging.

I propose a "copy and share" aggregation method to apply federated learning on Word2Vec models. Figure 5.4 shows example of the scheme for 2 participants. Firstly this 2 participants must agree on the same size of the hidden layer. They collect traffic for different IP addresses. Some of those senders are observed in both darknets and some are seen only in one of them. For those common sources the global model takes the average of the 2 participants and for the others the model just keeps the local value.

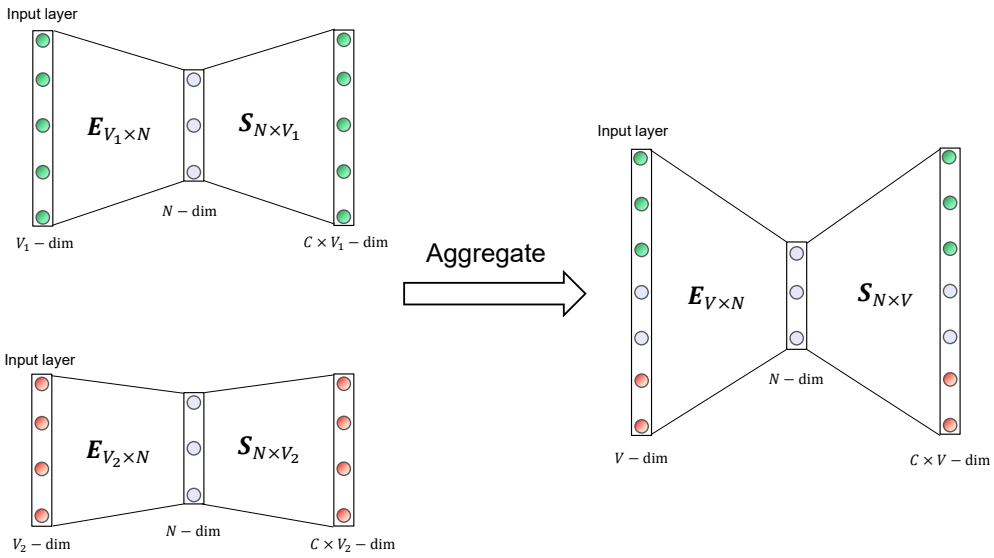


Figure 5.4: Model aggregation

If the number of participants increases, it will be more complex to find the intersection and averaging. To achieve better scalability, model aggregation is implemented in the following way: given K participants at time slot t, firstly the server extract a global vocabulary $\boldsymbol{V}$ from all participants k, i.e. $\boldsymbol{V} = \bigcup_{k=1}^{K} \boldsymbol{V}_k$; then

for each IP i in the vocabulary, if it is in the newest corpus $\boldsymbol{V}_{k,t-1}$,which means this IP address was active in darknet k in day $t-1$, a weight $w_{i,k,t}$ will be assigned, otherwise $w_{i,k,t}$ will be set to 0; not if i was not active in any participant darknets at $t-1$, its embedding will not be updated; finally the global embedding for this IP i is calculated as

$$E_{i,t} = \begin{cases} \frac{\sum_{k=1}^{K} w_{i,k,t} E_{i,k,t}}{\sum_{k=1}^{K} w_{i,k,t}} & \text{if } \exists k \in \{1,2..,K\}, i \in \boldsymbol{V}_{k,t-1} \\ E_{i,t-1} & \text{otherwise} \end{cases} \tag{5.1}$$

and the weight of hidden layer - output layer is computed with the same rule.

5.2.2 Weighting schemes

In a common federated learning scenario like FedAvg [15], the global model m is the weighted average of several participants as

$$m = \sum_{k=1}^{K} \frac{n_k}{n} m_k \tag{5.2}$$

where n_k is equal to the number of samples for training used by participant k to train the model and $n = \sum_{k=1}^{K} n_k$. However, in our case not all raw traces are used to train the model, and even in the local model, different IP addresses sending different numbers of packets will cause their embeddings to be updated at different frequencies. In order to adapt the problem, I design two weighting schemes.

Network-wise weighting A darknet observes more IP addresses means acquiring more information, so a higher weight is needed in model aggregation. A unified weight is applied to all IP addresses active in the last day $(t-1)$ in darknet k, and the weight for for each IP address i on day t is equal to the vocabulary size of the previous day:

$$w_{i,k,t} = |\boldsymbol{V}_{k,t-1}| \tag{5.3}$$

IP-wise weighting In the training process, embeddings for more active sources will be updated more frequently. If an IP address is very active in a darknet, then even though the total amount of traffic observed by this darknet is small, it may still extract a lot of information about this source. So I design this scheme, each IP address i corresponds to an individual weight, which is set to the number of packets it sent to the darknet k in the last day, denoted as $p_{i,k,t-1}$.

$$w_{i,k,t} = p_{i,k,t-1} \tag{5.4}$$

Chapter 6

Evaluation of federated learning solution

High-quality embeddings should have the ability to project senders with similar activity patterns into adjacent regions in the latent space. This allows us to evaluate the quality of IP embeddings by examining their neighborhoods. Typically, senders from the same source exhibit similar scanning activities, so their neighbors should have the same ground truth labels. In [1], a k-NN classifier was employed to assign labels to each IP address based on its k nearest neighbors using a majority voting rule and achieved high accuracy.

Following the same approach, I evaluate the embeddings of all IP addresses with ground truth using a 7-*NN classifier* and employ the *leave-one-out* approach. For each sample, the classifier is fit on the whole dataset, including samples labeled as "Unknown", while excluding itself. The majority class label among the k nearest neighbors will be assigned to the test sample. If the predicted label matches the true label, it indicates that the sample is correctly projected into a region surrounded by samples of the same ground truth class. So the accuracy of the classifier can reflect the quality of embeddings.

When evaluating the classification, for each class *True Positives* (TP) represent the instances correctly identified as belonging to that class, while *False Positives* (FP) are instances mistakenly identified as positive. *True Negatives* (TN) are instances correctly identified as not belonging to that class, and *False Negatives* (FN) are instances mistakenly identified as negative. These metrics help evaluate the accuracy and reliability of classification predictions. *Precision* and *Recall* can be defined with these matrics, precision for a class is fraction of correctly classified instances among all samples classified as this class, and recall is the fraction of samples that are recovered over all samples of a class.

$$Precision = \frac{TP}{TP + FP} \quad Recall = \frac{TP}{TP + FN} \tag{6.1}$$

Here I use *F1-score*, which is twice the harmonic mean of the precision and recall.

$$F1 = 2 \times \frac{Precision \times Recall}{Precision + Recall} \tag{6.2}$$

Given the fact that the number of samples included in the different classes is extremely unbalanced, I use *macro F1-score* when evaluating the overall performance, which is the unweighted mean of all the per-class F1 scores, otherwise the overall performance will highly biased towards the most populated classes.

And the distance metric used to define the nearest neighbors is *cosine distance*. For two row vectors $\mathbf{x}$ and $\mathbf{y}$, there consine distance $d_c(\mathbf{x}, \mathbf{y})$ is:

$$d_c(\mathbf{x}, \mathbf{y}) = 1 - \frac{\mathbf{x}\mathbf{y}^\intercal}{\|\mathbf{x}\| \, \|\mathbf{y}\|} \tag{6.3}$$

I train the embeddings with the 31 days traffic in May 2021, and the parameters of i-DarkVec are given in Table 6.1.

Parameter	Description	Value
m_{pkt}	Minimum number of packets send by an IP address	5
e	Dimension of embedding	200
c	Word2Vec context window size	5

Table 6.1: Parameters of i-DarkVec

6.1 Performance of different approaches

Firstly, it is necessary to evaluate whether collaborative training can enhance the quality of the embedding by transferring the knowledge extracted from different participant. To assess this, I conduct experiments using different collaborative training methods with involving two parties. For comparison, I evaluate these results alongside locally trained embeddings. Table 6.2 lists number of embeddings generated in different cases, where darknet 01 is the Polito /24 darknet and darknet 02 is a /24 subnet extracted from Brazilian /19 Darknet. Consider the whole dataset, about 40000 senders active in both darknets, which means their embeddings will be highly influenced by collaboration of two participants.

	Whole dataset (31 days)		Last day (05-31)	
	Samples	**GT Samples**	**Samples**	**GT Samples**
Darknet 01	74995	12947	9082	2314
Darknet 02	90097	13825	9451	2202
Global	127113	16389	13391	2856

Table 6.2: Comparison of number of embeddings

To facilitate comparison with locally trained samples, experiments are carried out by specifically selecting embeddings in the global model that correspond to IP addresses that are active in the network. It is important to highlight that *no weighting scheme* is employed in the federated learning approach discussed in this experiment. Therefore, during model aggregation, the weight $w_{i,k}$ for each IP address in all participants is uniformly set to 1.

Approach		**Darknet 01**	**Darknet 02**
Local	**31 days**	0.69	0.67
	Last day	0.76	0.83
Union	**31 days**	0.73	0.72
	Last day	0.76	0.82
Ping-pong	**31 days**	0.73	0.72
	Last day	0.77	0.83
Federated	**31 days**	0.71	0.71
	Last day	0.77	0.83

Table 6.3: Comparison of Macro F1-score with different approaches

The overall accuracy metric exhibits a strong bias towards the classes with large support, so I use macro averaged F1-score to show the overall results, results of the classification are listed in Table 6.3.

Figure 6.1 and Figure 6.2 illustrates the F1-score for each class. The F1-score for both darknets is positively impacted by collaborative training. Collaborative training either enhances or preserves the F1-score for all GT classes, with particular emphasis on classes such as "cybercasa" that demonstrate significantly improvement. Despite the usage of a simplistic weighting scheme, the embeddings generated by the federated learning approach still exhibit high quality.

Moreover, collaboration can also expand the coverage. During local training, certain observed senders were excluded due to insufficient traffic volume. However, they might be more active in another darknet. Thus, information shared by the

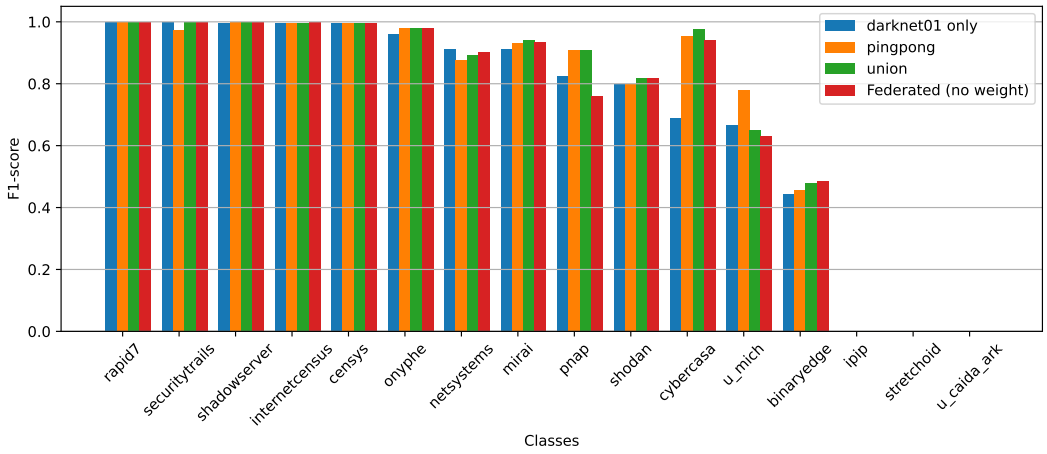


Figure 6.1: F1-score for all GT classes with in Darknet 01 (31 days)

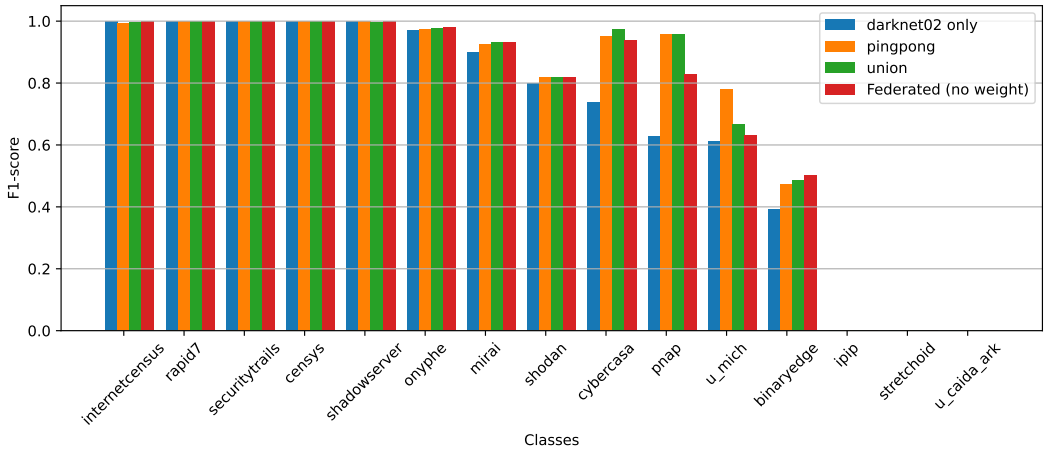


Figure 6.2: F1-score for all GT classes in Darknet 02 (31 days)

other participant enables the generation of embeddings for these senders. Table 6.4 shows the statistics for the coverage. *Extended coverage* refers to the number of IP addresses present in both darknets but only active in the other one. with only one collaborator of the same size, federated learning approach allows darknet 01 to extend 14.7% of its coverage, and darknet 02 extends 12.3%.

	Observed senders	IP embeddings	Extended coverage
Darknet 01	532609	79445	11735
Darknet 02	591213	90097	11038

Table 6.4: Coverage of IP embeddings for two darknets

6.2 Performance of different weighting schemes

The results presented in Section 6.1 confirm the assumption that collaborative training can facilitate the transfer of information extracted from various darknets, thereby enhancing the quality of IP embeddings. To further stress the performance of federated learning approach, I conducted experiments to assess the impact of weighting schemes. These experiments encompassed all embeddings in the global model and the union approach, which closely resembles a centralized dataset scenario, serves as the reference for comparison.

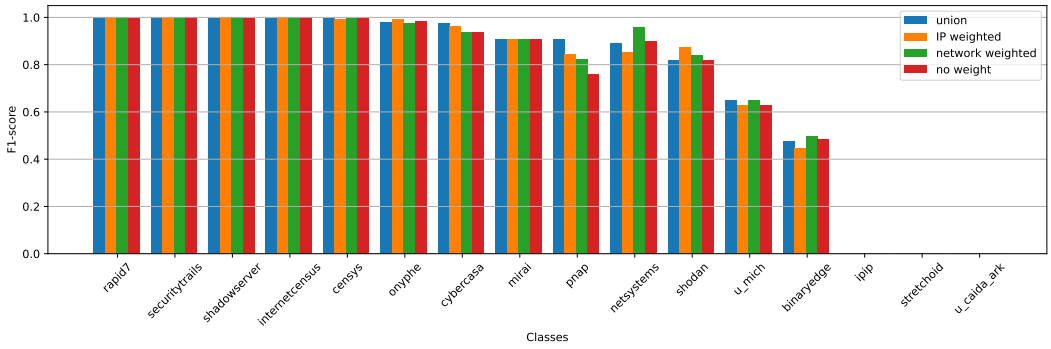


Figure 6.3: F1-score for all GT classes with different weighting schemes

Figure 6.3 demonstrates the F1-score for each class, there is no significant difference in results for most classes, the performance slight improved after applying weighting, for all classes at least one of the two weighting schemes achieves better F1-score than the case without weight. Figure 6.4 reports the distribution of the F1-Scores, as shown in the boxplot, the network-wise weighting achieves better performance than the IP-wise weighting. In this case the macro F1-score with network-wise weighting is 0.72, almost same as the union.

6.3 Performance in different scenarios

After the initial testing of federated learning between two /24 darknets, the subsequent experiments aim to assess the performance of federated learning approach in

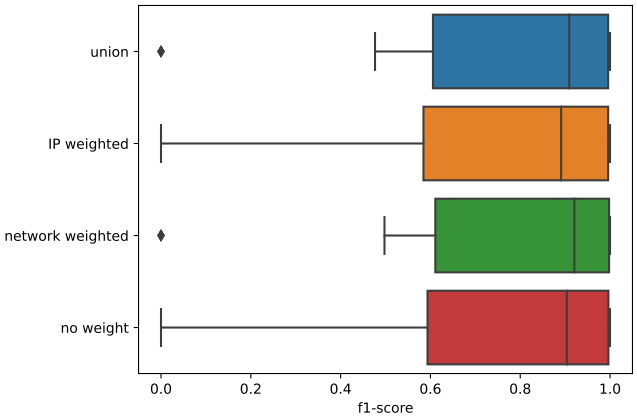


Figure 6.4: F1-score distributions with different weighting schemes

more complex scenarios.

6.3.1 Heterogeneous networks

The purpose of this experiment is to investigate the performance of the federated learning approach when involving participants from two darknets of varying sizes. In this experiment, I still designate the Polito /24 darknet as the darknet 01, and select a variable range of IP addresses from the Brazilian darknet as darknet 02. The size of darknet 02 varies from /20 to /28.

Similar to experiments shown in Section 6.1, for each darknet, evaluation is performed on IP addresses that are active in it and compared to embeddings trained with union approach and with only local traffic.

Figure 6.5 shows the macro F1-score of darknet 02. It is obvious that when darknet 02 comprises fewer than 256 IP addresses, the performance experiences a significant decline due to insufficient traffic collection. However, leveraging the information extracted from darknet 01 allows darknet 02 to maintain a certain level of quality of its IP embeddings. This is more pronounced when darknet 02 becomes smaller, in this scenario the weight of darknet 02 drops to a very low level and the global model is mostly contributed by darknet 01. Consider darknet 02 can collect quite limited traffic, the senders still active in darknet 02 are those IP addresses that send packets more frequently and regularly, and daknet 01 can generate high quality embeddings for them, so the F1 score increases a bit.

Figure 6.6 shows that the macro F1-score of darknet 01 remains relatively stable despite variations in the size of darknet 02. During federated learning, if darknet 01 collaborates with a darknet that is equal to or larger in size, it can benefit from the information shared by the other. In the case of darknet 02 being a small network,

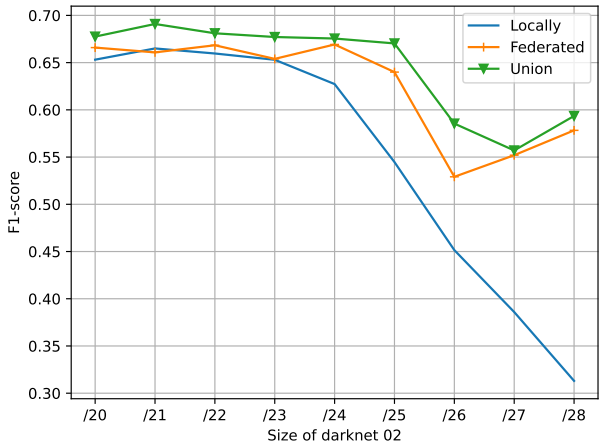


Figure 6.5: Macro F1-score of darknet 02

it lacks the capability to train a proficient local model. In a federated learning scenario, averaging its embeddings may introduce noise to the global model, leading to a slight decrease in the F1-score to even worse than local case. However, this issue does not exist in the union scenario.

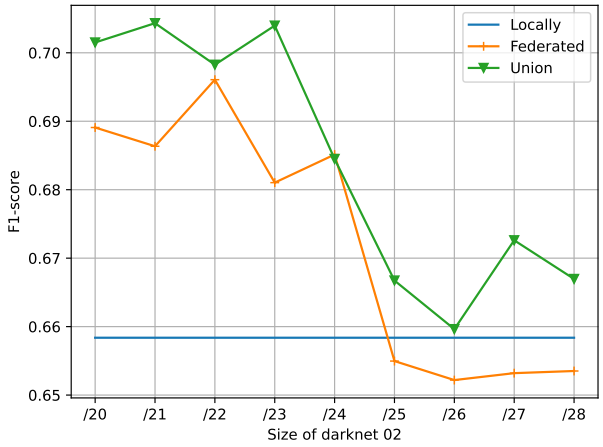


Figure 6.6: Macro F1-score of darknet 01

The results from these two darknets uncover a phenomenon within the federated learning approach. It demonstrates that both parties can benefit from federated learning when they both possess a sufficient amount of data. However, when one participant lacks sufficient data or when there is a significant disparity in data amounts between the two parties, the activities of the senders observed by the darknet with a smaller data volume are very limited and has a large gap with the

larger party. When averaged with representations generated from uncomprehensive observations, the larger participant may be slightly "polluted".

Regarding coverage, the participant with a smaller data quantity will significantly benefit from federated learning, whereas the other side will only be able to extend minimal coverage. For instance, consider a /24 darknet that gets support from a /20 darknet, the number of sender IP addresses it can cover grows from 79.4K to 303.1K, nearly four times its original number. but adding the information of a /24 Network to a /20 network would only increase the coverage of about 0.08%, as it can be better seen in the rightmost part of Figure 6.7 and Figure 6.8.

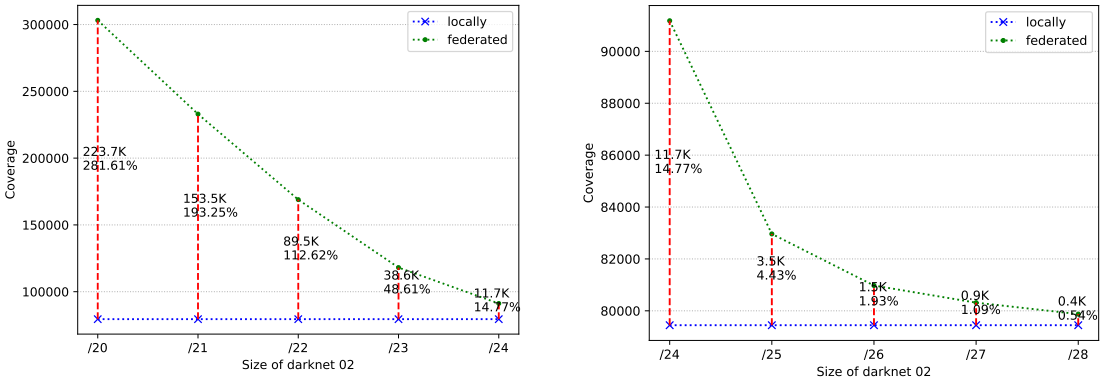


Figure 6.7: Coverage extension in Darknet 01

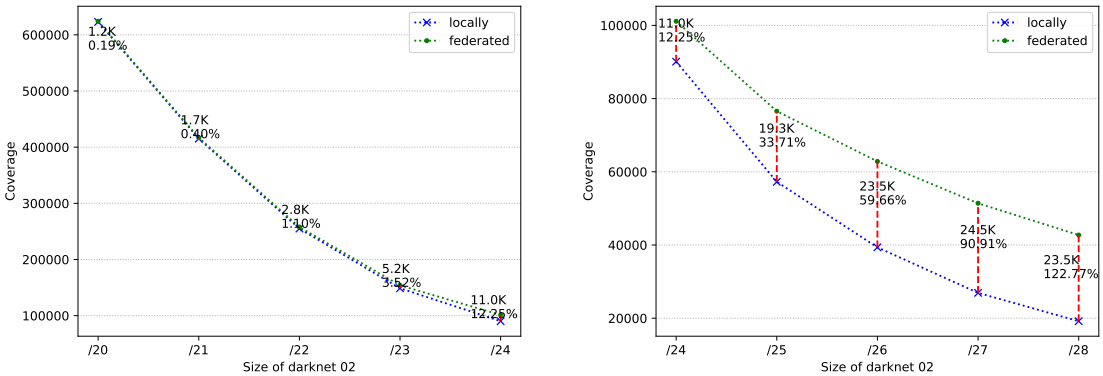


Figure 6.8: Coverage extension in Darknet 02

6.3.2 Multiple networks

A significant advantage of the federated learning solution over other simplistic methods is its scalability. Therefore, it is crucial to assess its performance as the number of participants increases.

In this experiment all participant darknets are extracted from Brazilian /19 darknet, I selected various subnet that separate from each other. The 7-NN classifiers are trained with the global embeddings. Figure 6.9 shows the coverage of global model (vocabulary size) in various scenarios. As the number of participants in the federated learning increases, the coverage expands significantly.

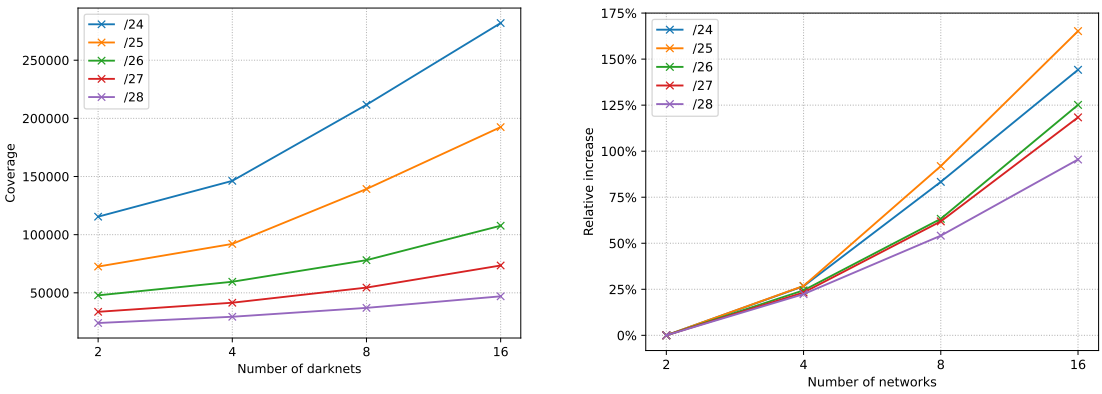


Figure 6.9: Coverage of multiple darknets

Figure 6.10 and Figure 6.11 illustrate the performance with different number of darknets. when participants possess sufficient data to generate meaningful embeddings, the quality of these embeddings can be enhanced with an increasing number of participants. Conversely, if participants can only generate noisy embeddings, the quality of the global model will not improve, regardless of the number of participants involved.

6.4 Summary

The results of experiments presented in Section 6.1 demonstrate the feasibility of applying federated learning in darknet traffic analysis. With appropriate settings, in the simple collaboration scenario, the federated learning solution does improve the quality of generated embeddings compared to the local case. In fact, it exhibits strong performance that closely resembles centralizing data from multiple parties.

However, the current federated learning solution exhibits limited performance in more complex scenarios. One notable limitation is its inability to effectively

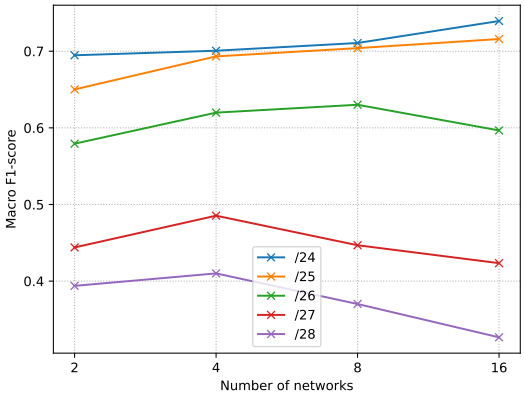


Figure 6.10: Macro F1-scores.

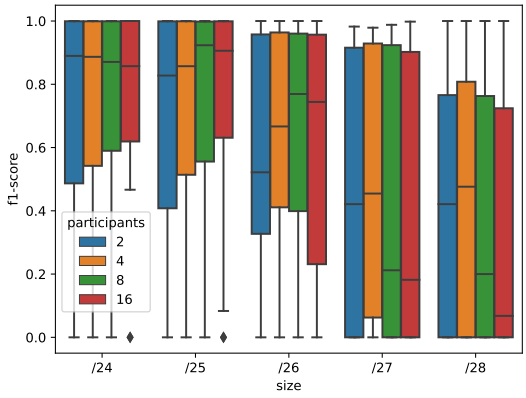


Figure 6.11: F1-scores distributions.

handle unbalanced participant contributions. When there is a significant disparity in the sizes of the participants, the performance of the federated learning solution deviates further from the centralized case or may even be inferior to the local case. Additionally, if none of the participants collect a sufficient amount of traffic for training, the results obtained, despite involving a large number of participants, remain unsatisfactory. Handling these limitations can be considered as future work.

Another benefit of federated learning is that it extends coverage. By participating in federated learning, each party, even if it is much larger than others, can generate embeddings for some senders that were previously filtered out during corpus generation.

Chapter 7

Monitoring clusters evolution

Clustering methods can identify senders with similar activity patterns on that day out, pointing out some coordinated activities. These coordinated activities may change over time, and with the incremental training the embeddings can reflect the newest activity patterns, thus resulting in different clusters. Monitoring the evolution of clustering results at different times can extract more information. To achieve this objective, it is necessary to employ multiple metrics for evaluating the clustering outcomes.

7.1 Conventional clustering evaluation metrics

Several traditional measures are used to assess the clustering results, including extrinsic metrics that draw upon ground truth labels for reference. These metrics are implemented to quantify the degree of similarity between two distinct clustering outcomes: the clustering on a designated day, representing the ground truth, and the clustering result from another day, serving as the projected outcome.

The following extrinsic metrics are employed:

- **1. Adjusted Rand Index (ARI)** Rand Index (RI) considers all pairs of samples and compares their clustering assignments to determine the agreement between the two clusterings. X and Y are two different clustering, RI is given by

$$RI = \frac{a+b}{\binom{n}{2}} \tag{7.1}$$

where a is the number of pairs of elements that are in the same cluster in X and in the same cluster in Y, b is the number of pairs of elements that are in

different clusters in X and in different clusters in Y. ARI is defined as

$$ARI = \frac{RI - \mathbf{E}[RI]}{max(RI) - \mathbf{E}[RI]} \tag{7.2}$$

where $\mathbf{E}[RI]$ is the expected RI of random clustering, and $max(RI)$ represents the maximum possible value of RI. The ARI value ranges from -1 to 1, where a value of 1 indicates a perfect match between the two clusterings, 0 indicates random agreement, and -1 indicates complete disagreement.

- **2. Normalized Mutual Information (NMI)** NMI considers entropy and size of clusters, for two different clustering outcomes x and Y NMI is computed as

$$NMI = \frac{2 \times \mathbf{I}(X, Y)}{[\mathbf{H}(X) + \mathbf{H}(Y)]} \tag{7.3}$$

where $\mathbf{H}(.)$ denotes the entropy and $\mathbf{I}(X, Y))$ is the mutual information between X and Y. NMI provides values between 0 and 1, where a value of 1 indicates a perfect match between the clusterings, and 0 indicates no similarity.

In addition to the aforementioned metrics, the intrinsic metric, **silhouette** (sh), is also employed to assess the clustering. Unlike extrinsic metrics, the silhouette metric does not require any external information. This measure evaluates the fit of each data point within its designated cluster relative to other clusters, thereby providing insights into the degree of separation and compactness of the clusters. The silhouette value ranges from -1 to 1. A value approaching 1 signifies well-separated clusters, values around 0 suggest overlapping clusters, while negative values may indicate the potential misassignment of data points to incorrect clusters. For a single sample i the silhouette coefficient $s(i)$ is then given as:

$$s(i) = \frac{b(i) - a(i))}{max(b(i), a(i))} \tag{7.4}$$

where $a(i)$ is the mean distance between i and all other samples in the same cluster, $b(i)$ is the mean distance between i and and all other points in the next nearest cluster. The evaluation of clusters is facilitated using the average silhouette. For a given cluster C, its average silhouette is computed as follows:

$$s(C) = \sum_{i \in C} \frac{1}{|C|} s(i) \tag{7.5}$$

7.2 MONIC method

MONIC[55] is a framework designed to model and track cluster transitions. It takes clustering outcomes from consecutive time points as input and employs a

transition model that covers changes involving multiple clusters. Consequently, it offers valuable insights into the overall cluster dynamics. To align the MONIC method with the IP embedding clusters scenario, certain modifications have been made to enhance its suitability. This adaptation accounts for a high proportion of samples being clustered as noise and the significant variation in samples for clustering across different timeslots.

MONIC is more concerned with the changes in members of clusters than their spatial properties, thus cluster overlap is a fundamental building block of MONIC. The overlap of cluster X and another cluster Y is defined as

$$OL(X,Y) = \frac{|X \cap Y|}{|X|} \tag{7.6}$$

Given clustering outcomes of a time point t_0 and a subsequent time point t_1, denoted as ζ_0 and ζ_1, various clusters transitions can be tracked according to the overlap of clusters in different clustering outcomes. Let X_i be a cluster in ζ_0 and Y_i be a cluster in ζ_1, when $i = -1$ the cluster represents the noise samples. Some senders active in t_0 may not be active in the following time point, and it is necessary to consider how many members of a cluster is still active in t_1, Equation 7.7 shows the definition of active fraction.

$$A(X) = \sum_{Y \in \zeta_1} overlap(X,Y) \tag{7.7}$$

Transitions of a X_i are defined in Table 7.1, where $\tau_0 \in [0.5,1]$ and $\tau_1 \in [0, 0.5)$ are thresholds. If $OL(X,Y)/A(X) \geq \tau_0$, then X matches Y. It should be noted that the transition of the noise sample cluster X_{-1} is not taken into consideration.

And for cluster Y_j in ζ_1, there are some rules to identify if it is *emerged*:

- There is no cluster in t_0 matches Y_j.

$$\forall X_i \in \zeta_0, i \neq -1: \quad OL(X_i, Y_j)/A(X_i) < \tau_0 \tag{7.8}$$

  In this case, Y_j can represent a cluster mainly composed of newly observed IP addresses or predominantly originating from noise, among other possibilities.

- There are more than one cluster in t_0 matches Y_j, but none of them is a major component of Y_j.

$$\begin{aligned} \exists X_i \in \zeta_0, i \neq -1: \quad & OL(X_i, Y_j)/A(X_i) \geq \tau_0 \\ \exists X_k \in \zeta_0, k \neq -1, k \neq i: \quad & OL(X_k, Y_j)/A(X_k) \geq \tau_0 \\ \forall X_i \in \zeta_0, i \neq -1: \quad & OL(Y_j, X_i) < \tau_0 \end{aligned} \tag{7.9}$$

  In this scenario, Y_j can be a large cluster that absorbs a lot of small clusters, a cluster formed by merging a few clusters of approximately equal size or other potential conditions.

Transition	**Formulation**
Survived	$\exists Y_j \in \zeta_1, j \neq -1 : \quad OL(X_i, Y_j)/A(X_i) \geq \tau_0$ $\nexists k \neq i : OL(X_k, Y_j)/A(X_k) \geq \tau_0$ OR $OL(Y_j, X_k) \geq \tau_0$ $A(X_i) \geq \tau_1$
Fragmented	$\forall Y_j \in \zeta_1, j \neq -1 : \quad OL(X_i, Y_j)/A(X_i) < \tau_1$ $OL(X_i, Y_{-1})/A(X_i) < \tau_0$ $A(X_i) \geq \tau_1$
Split	$\forall Y_j \in \zeta_1 \quad OL(X_i, Y_j)/A(X_i) < \tau_0$ $\exists Y_j \in \zeta_1, j \neq -1 : \quad OL(X_i, Y_j)/A(X_i) \geq \tau_1$ $A(X_i) \geq \tau_1$
Absorbed	$\exists Y_j \in \zeta_1, j \neq -1 : \quad OL(X_i, Y_j)/A(X_i) \geq \tau_0$ $OL(Y_j, X_k) < \tau_0$ $\exists k \neq i, j \neq -1 : \quad OL(X_k, Y_j)/A(X_k) \geq \tau_0$ $A(X_i) \geq \tau_1$
Disappeared	$OL(X_i, Y_{-1})/A(X_i) \geq \tau_0$ $A(X_i) \geq \tau_1$
Inactive	$A(X_i) < \tau_1$

Table 7.1: Transitions of a cluster X_i

Chapter 8

Results of clustering evolution

The experiments for monitoring cluster evolution are conducted using Polito /24 darknet traffic. The process is illustrated in Figure 8.1. Robust embeddings are required for performing clustering. Initially, I train embeddings with i-DarkVec for 31 days, and subsequently, I apply HDBSCAN on these embeddings corresponding to IP addresses active on the last day. The embeddings are updated on the followings day based on the newly observed traffic, and then clustering is repeated on the new embeddings. This procedure is repeated for 20 days, and the 20 clustering results are used to monitor the evolution. The hyper parameter m_{pkts} is always set to 10.

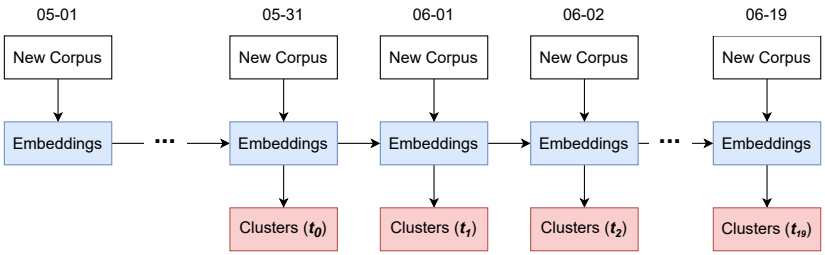


Figure 8.1: Training procedure of clustering

8.1 Clustering results overview

During this 20-day period, more than 52000 distinct active IP addresses are clustered. Figure 8.2 shows the number of active senders in each considered date. Approximately seven to nine thousand IP addresses can be clustered each day,

of which one to two thousand are newly observed sources. These numbers do not change dramatically, which means that no internet-scale large network events like Mirai botnets spread in September 2016 [4] occurred during the 20 days of observation.

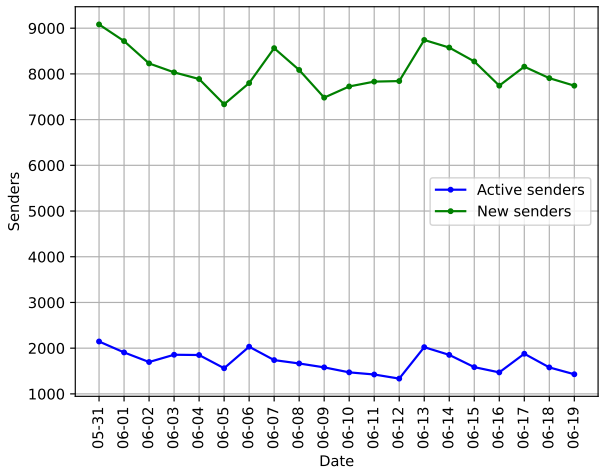
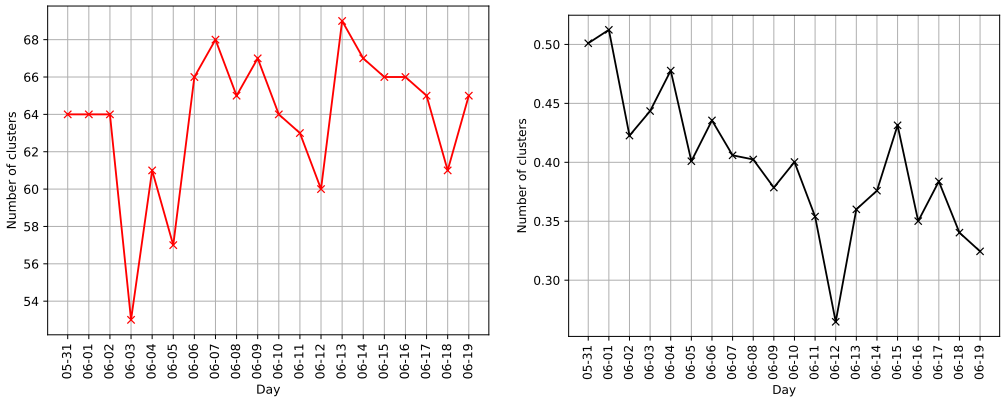


Figure 8.2: Active senders on specific date.

Figure 8.3 presents the basic characteristics of the clustering results over the investigated 20 days. Each day, approximately fifty to seventy clusters can be extracted, while a significant proportion of the active IP addresses are identified as noise.



(a) Number of clusters **(b)** Ratio of noise samples

Figure 8.3: Basic characteristics of the clustering results.

Upon further investigation of the noise samples, it was found that 33685 senders were clustered into the noise at least once over the 20-day period. Figure 8.4 depicts the distribution of the number of days these senders were active and the number of days they were identified as noise. Over 50% of these IP addresses were active in the darknet for only 1 day and were identified as noise, it is likely that these sources accessed the darknet randomly, possibly due to misconfiguration or other factors. While those senders who are consistently active in the darknet are rarely defined as noise all the time, more than 4000 senders were observed only once in noisy clusters, whereas they were active for more than one day. The presence of these senders in the noise could be attributed to the randomness of the clustering algorithm.

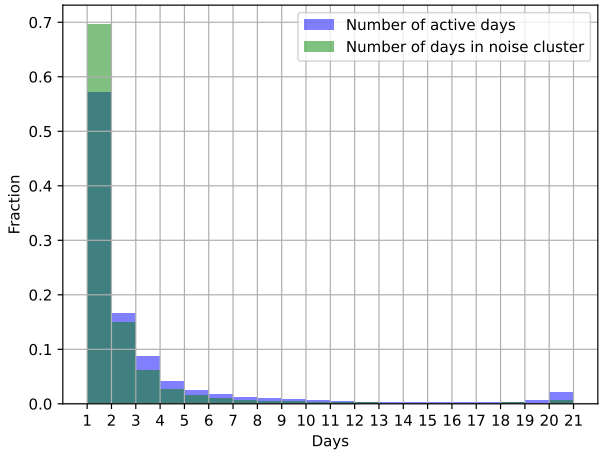


Figure 8.4: Senders in noise.

When examining the extracted clusters, the repeated clustering over the 20-day period produced more than 1200 clusters, the majority of which exhibited high average silhouette values. As shown in Figure 8.5, there are more than 1000 clusters with exceeding 0.6, indicating a good separation among clusters and implying the senders are well-clustered.

As expected, the profile of the clustering results is consistent with the nature of HDBSCAN, effectively separating outliers and extracting significant clusters.

The cluster can be labeled based on the most prevalent class when combining samples within it with the ground truth. For example if the majority of samples in a cluster are of unknown source, then this cluster will also be labeled as location. The proportion of samples belonging to the most prevalent class within the cluster can be defined as the **purity**. As shown in Figure 8.6, throughout the 20-day period, there are over 800 clusters that the majority of their samples originating from known sources, generally clusters with better silhouette value has higher

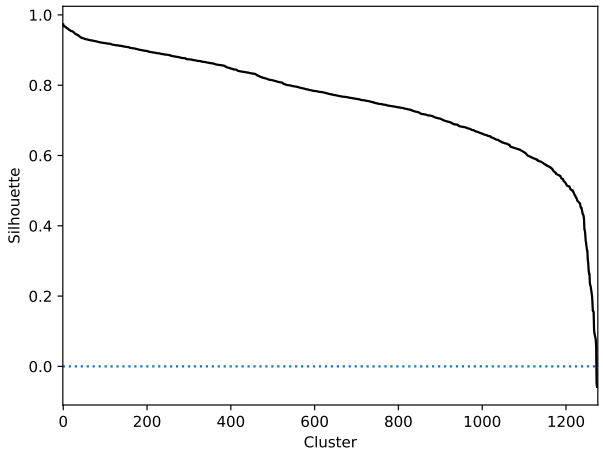
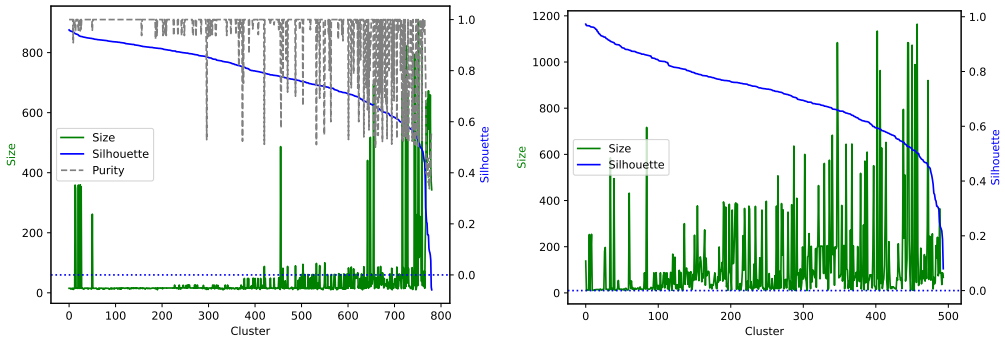


Figure 8.5: Silhouette of all clusters.

purity. For unknown clusters, the purity is not very meaningful because there is no way to verify whether the samples belong to a known ground truth class. In both cases, huge clusters trend to have lower silhouette.



(a) Clusters with known classes **(b)** Clusters labeled as unknown

Figure 8.6: Silhouette, size of labeled and unknown clusters.

8.2 Clusters evolution

The experiment for monitoring cluster evolution involved comparing the cluster results of each day with those of the previous days, aiming to identify changes in the cluster by observing their similarity.

8.2.1 Conventional metrics

Given the fact that each day the active senders are not exactly the same as the other dates, it is necessary to extract the senders active on both days when comparing clustering results of two different days. Figure 8.7 shows the active senders that overlap on different dates. By taking into account the number of active senders per day (Figure 8.2), it can be observed that each day comprises roughly half of the senders who were active on the previous day. In comparison to the initial day of clustering, the overlap of active senders diminishes over time, eventually encompassing only those that exhibit consistent activity.

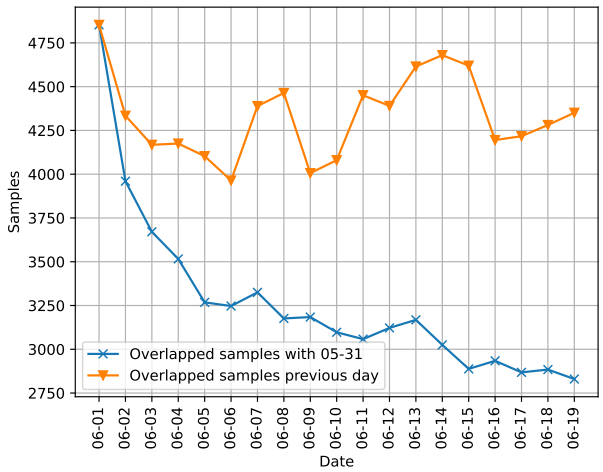
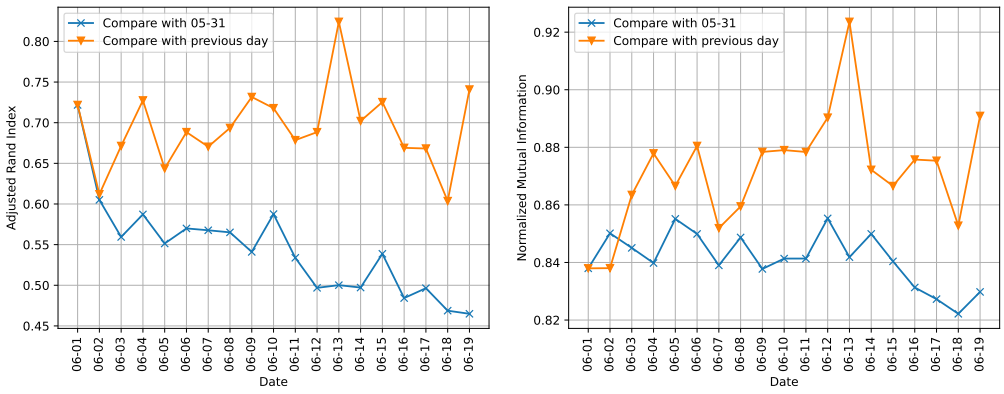


Figure 8.7: Overlapped active senders.

Figure 8.8 shows the result of the ARI and NMI in comparison of two clustering results with the overlapped senders. The ARI does not show a very regular change when compared with the previous day. When making a comparison with the first day, the value of ARI showed a decreasing trend with a longer time interval, indicating that the dissimilarity of clustering results is accumulating over time as embeddings being updated continuously. The process of calculating the NMI value takes into account the size of the clusterings and normalized by the entropy of the two clusterings, resulting in a higher score than the ARI. The magnitude of change in NMI during the 20-day period of observation is smaller, and there is also no significant decrease over time when compared to the first day.

The ARI and NMI values are relatively high, suggesting that the clustering results have remained stable compared to the previous day. It is important to note that the calculation of NMI and ARI incorporates noise, which accounts for instances where a sample is clustered on one day but considered an outlier on another day, reflecting a change.



(a) Adjusted Rand Index **(b)** Normalized Mutual Information

Figure 8.8: Silhouette, size of labeled and unknown clusters.

Moreover, the average silhouette for the entire clustering set was evaluated for each date. As noisy clusters, encompassing outliers, can be randomly distributed within the latent space, including these samples in the assessment would be meaningless. Hence, they were excluded from the silhouette calculation. Figure 8.9 shows the result, the values mostly distributed between 0.6 and 0.7, no particularly dramatic changes.However with only the average silhouette value is difficult to know what happened in those clusters.

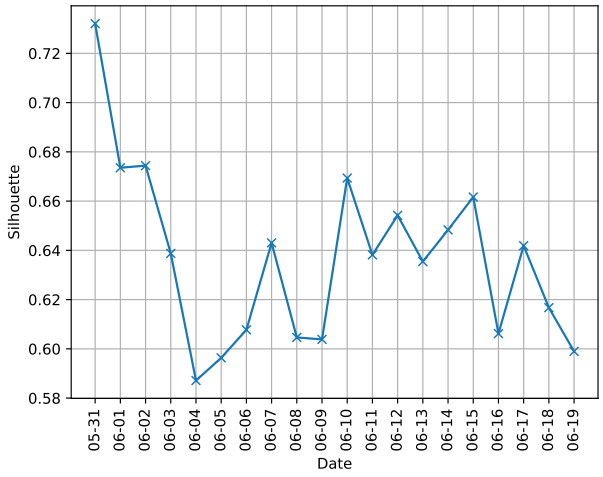


Figure 8.9: Average silhouette of each day.

In summary, traditional metrics, both intrinsic and extrinsic, can only offer very limited information. They mostly indicate whether there are significant changes in consistently active senders from a general perspective. However, they lack

information about new activities and changes within specific clusters.

8.2.2 MONIC method

MONIC method monitors clusters transitions based on the overlap (Equation 7.6) of two clusters in different clusterings and two threshold τ_0 and τ_1. Here I set $\tau_0 = 0.65$ and $\tau_1 = 0.3$.

Figure 8.10 shows the distribution of each cluster's maximum overlap to next day over its active fraction ($max\frac{OL(X,Y)}{A(X)}$, maximum active overlap for short) in the whole 20-day period, note if all senders in a cluster are not active in the following day, the maximum active overlap value set to 0. Most of clusters has a maximum active overlap greater than 0.9, means most of their members are still active in the same cluster on the following day, it is expected that most of these clusters will be observed to survived.

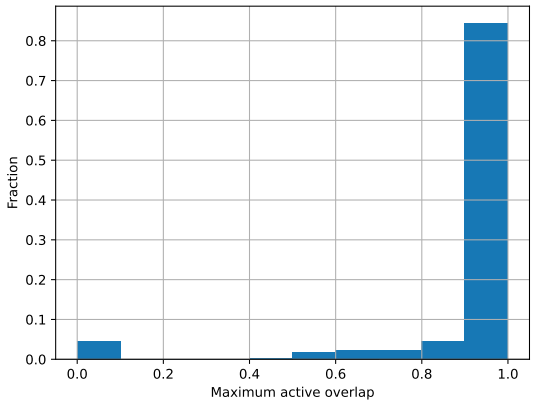


Figure 8.10: Maximum active overlap in 20-day period.

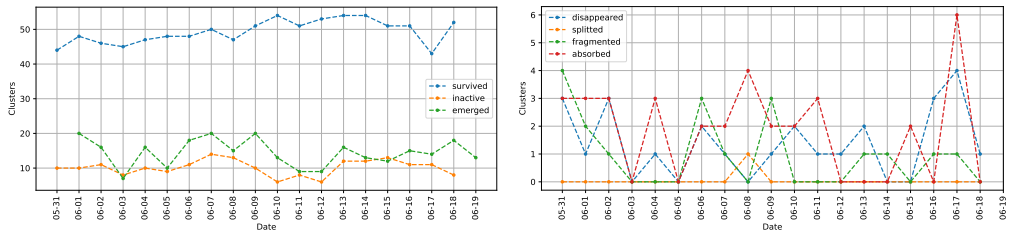
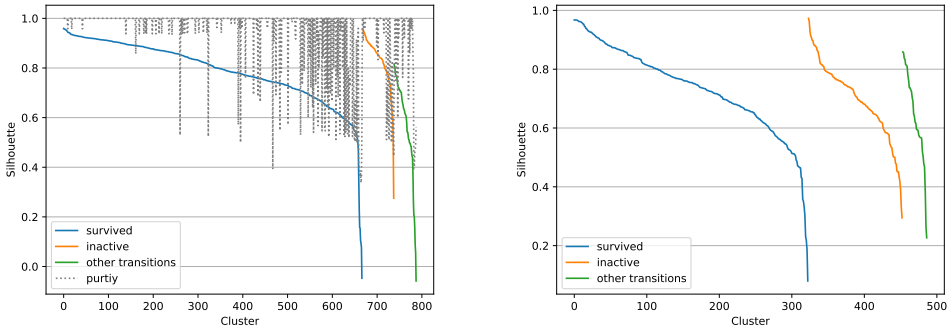


Figure 8.11: Clustering evolution trend in the 20-day period.

Figure 8.11 shows the number of clusters corresponding to various types of transitions for each day. It is evident that the majority of the clustering are

survived, suggesting that these groups of senders will still exhibit a similar activity pattern on the subsequent day. And 10-20 clusters will emerge each day, and close to 10 clusters become inactive on the next day. A smaller number of clusters experienced alternative transitions, typically fewer than 5 or even 0 per day.

Figure 8.11 shows silhouette of clusters experienced different transitions. If a cluster is still active but does not persist on the following day, usually it will not have a very high silhouette value,also low purity is more common in such clusters,



(a) Clusters with known classes **(b)** Clusters labeled as unknown

Figure 8.12: Clusters experienced different transitions.

For emerged clusters, it is important to check where their members come from, Figure 8.13 displays the distribution of the proportion of new IP addresses within these clusters, along with the distribution of the proportion of members come from noise.In this context, a "new" IP address can refer to one that has never been observed before or a known sender that was inactive the previous day.

During this 20-day period, very few clusters emerged mainly from the noise of the previous day since it's hard to see some active senders with irregular activities being coordinated at some point. Over half of the emerged clusters are mainly composed of new IP addresses, which can be attributed to periodic activities (e.g., scanning every few days) or newly discovered activities. In fact more than 85% of these emerged clusters survive or become inactive in the next day. The monitoring of emerged clusters offers the possibility to detect new activities and identify changes in previously observed periodic activities.

8.3 Tracking clusters

Given the fact most of clusters identified survived, it is possible to track their evolution during the period and extract more information about the activities

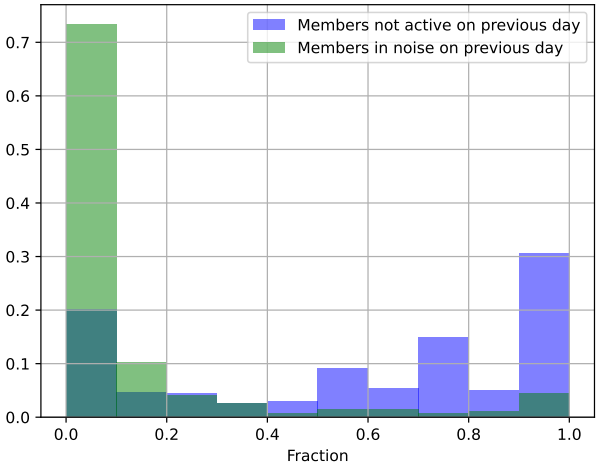
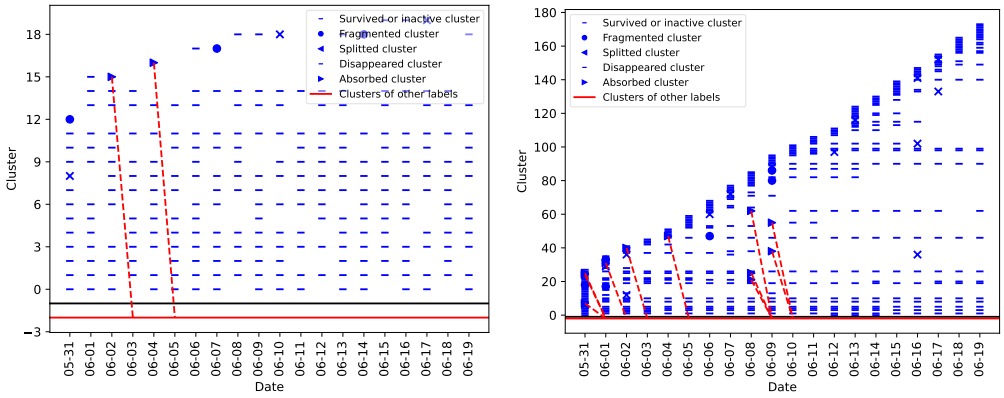


Figure 8.13: Origin of the members in emerged clusters.

corresponding to these clusters. Figure 8.14 shows the examples of censys clusters and unknown clusters.



(a) Censys **(b)** Unknown

Figure 8.14: Clusters evolution.

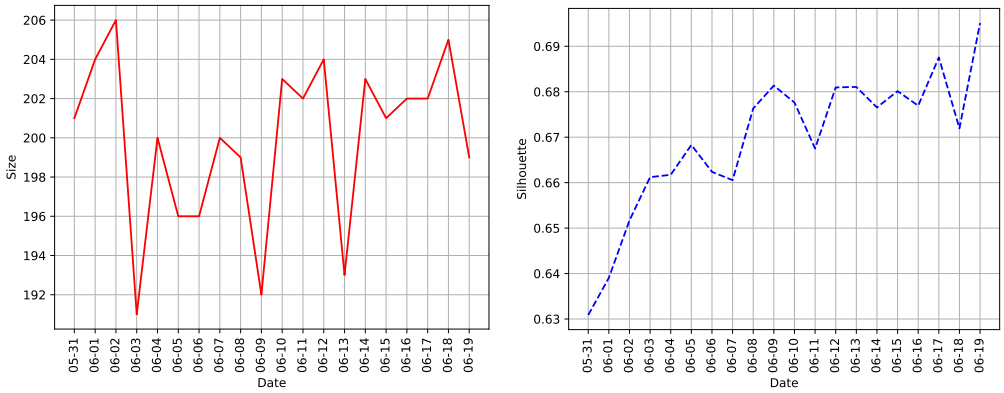
As shown in Figure 8.14b, even though new unknown source clusters are detected everyday, most of them did not persist for very long period. Whereas clusters compose of known sources, like the example for Censys in Figure 8.14a, are constantly persist since their are constantly perform samilar scanning activities, and due to the limited ground truth almost no new labeled clusters can be detected. The phenomena explains why during the 20-period the number of clusters with known clusters is nearly twice as large as the number of unknown source clusters, same group of labeled senders are clustered almost every day.

Focusing on unknown clusters is more important for extracting meaningful insights due to limited prior knowledge about the senders. Inspecting these clusters provides opportunities to uncover valuable information such as unknown malware and potential attacks.

Given that the majority of unknown clusters vanished shortly after their emergence, those that persist for a longer duration hold particular significance. Some use cases are described in the following sections.

8.3.1 Use case: Shadowserver stable cluster

Cluster 5 in Figure 8.14b survived for 20 days, and it was quite stable in the investigated period. As Figure 8.15a shows, its size almost unchanged at around 200 and During the 20-day period, a total of 216 IP addresses appear in this cluster, which means there is little change in its members and the activity of these senders was very regular. Its silhouette value also varies little, and is generally increasing.



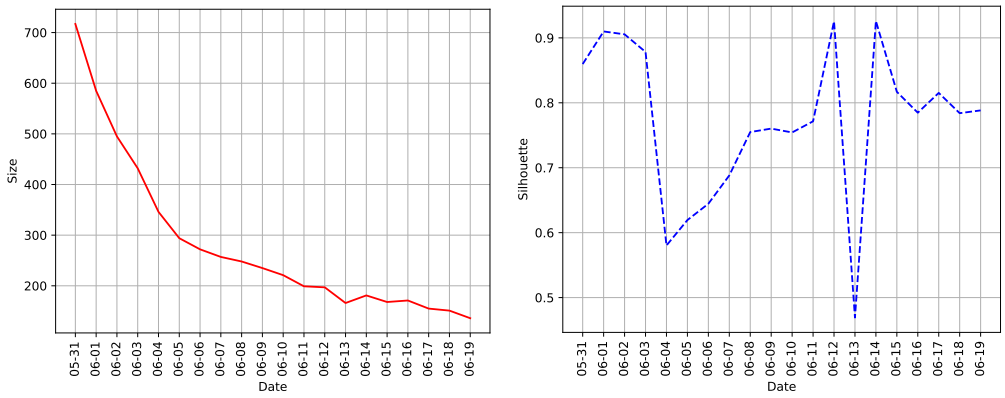
(a) Size **(b)** Silhouette

Figure 8.15: Properties of cluster 5.

When mannuly checking the property of these IP addresses, 210 IP addresses are from the same /24 subnet, located in White Salmon in United States and 5 IP addresses are from another /24 subnet in another US city Stevensville. Their autonomous systems are all organized by a US network provider Hurricane. And according to the result of reverse DNS lookup, their domain belongs to shadowserver, which is a class in the ground truth. It is pretty evident that these senders are from this known source, but they are not included in our labels. By inspecting this cluster we can extend our knowledge about the ground truth.

8.3.2 Identify cyber incidents

Another cluster 27 also suvived for 20 days, but the members of this cluster are not so stable, as shown in Figure 8.16, it keeps shrinking over time, on May 31 it contained 717 IP addresses, while 20 days later there were only 136, its silhouette value also varies a lot.

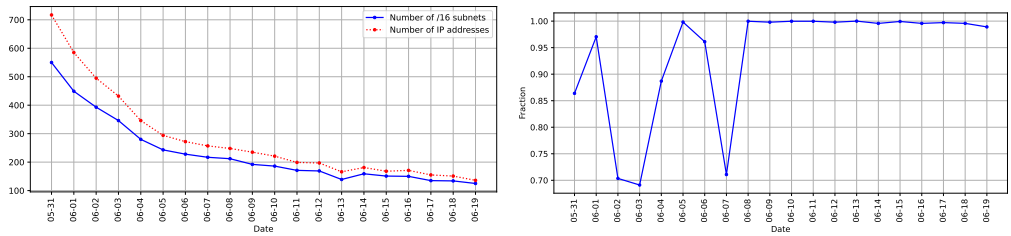


(a) Size **(b)** Silhouette

Figure 8.16: Properties of cluster 27.

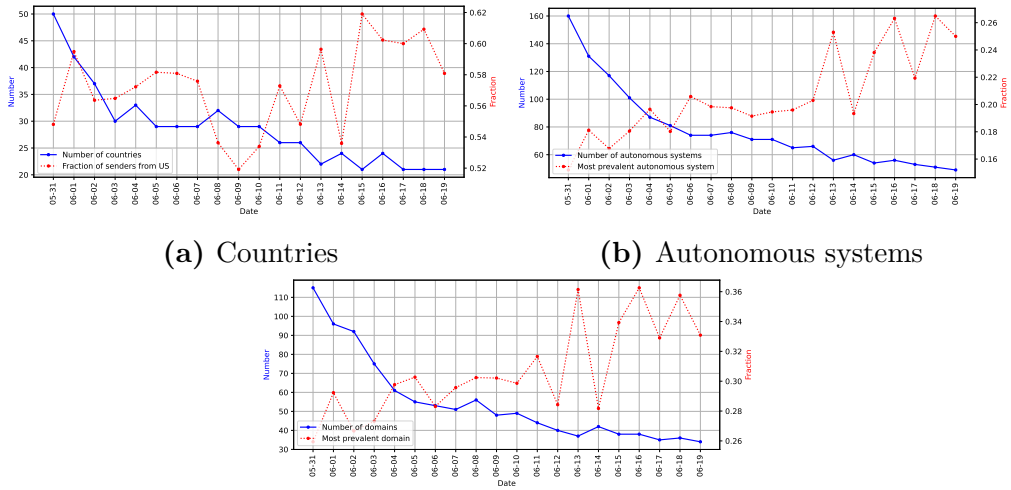
To characterize the cluster, it is necessary to further inspect it. Firstly is the distribution of IP addresses, in this cluster sources are scattered all over the IP space, they are from different /16 subnets, the most prevalent /16 subnet in this cluster only contains less than 4% of senders. With the exception of three days, the majority of the traffic consists of data packets using the GRE protocol. These three days exhibited similarities as only a few IP addresses were responsible for sending a significant number of packets in other protocol. On June 2, approximately 20 newly observed IP addresses joined the cluster and sent TCP traffic. Some of these IP addresses remained active in the cluster the next day, but they all became inactive on June 4. On June 13, there were only 2 senders who transmitted 3.8k UDP packets. As a comparison, in this cluster a sender sends an average of about 20 packets per day.

As for the geographic locations, they are located in different countries, more than half of them are in US. These IP addresses also organized by different autonomous systems, and the most prevelant autonomous system is "TWC-20001-PACWEST" operated by Charter Communications in US. The result of reverse DNS lookup shows that these IP addresses come from different domains, the most frequent of



(a) IP addresses distribution **(b)** Fraction of GRE packets

Figure 8.17: Other properties of cluster 27.



(a) Countries **(b)** Autonomous systems

(c) Domains

Figure 8.18: External information about cluster 27.

which is "rr.com", a US email service, which was rebrand as "specturm"[1]. And the domain name "specturm.com" also appears in the cluster with a lower popularity.

Figure 8.19 shows the activity patterns of senders in this cluster on May 31. Their activities in the previous days are also included. The vast majority of them were inactive until May 30, after which they became extremely active in the week following while later there are gradually start to stop. In fact as mentioned in Section 4.1, the GRE packets are rarely collected in the first 30 days of May, confirmed that this is not a common event, GRE protocol are commonly used in

[1]https://help.infusionsoft.com/help/roadrunner-addresses-disabled-from-sending

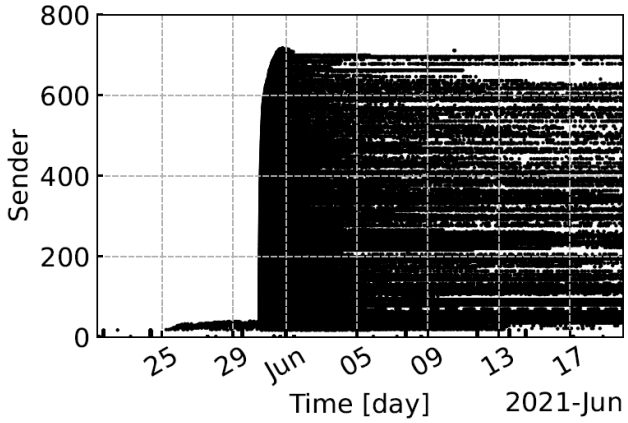


Figure 8.19: Activity patterns of cluster 27.

anti-DDoS sytems[2], and also can be used to carry DDoS attack[3]. It can be iferred that this incident is related to an attack. Considering the volume of received traffic and given the fact these sender vanished as time passes, these senders are most likely to be victims of the attack and so the traffic is backscattering, presumably those IP addresses in US are the primary attack targets.

8.3.3 Other clusters

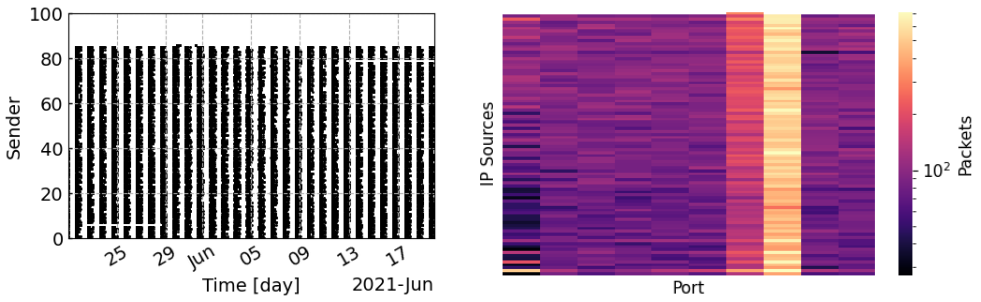
Several other clusters of senders, previously unlabeled and persistently active, are presented. Determining their definitive intent poses a challenge, but some key attributes of these groups can be identified.

Cluster 1 is a small cluster consisting of less than 20 IP addresses that constantly send packets targeting TCP port 8545 over the 20-day period. The reverse DNS lookup queries cannot return valid domain name. These source IP addresses are scattered across different /16 networks. TCP port 8545 is used for Remote Procedure Call and usually used as the default API-endpoint among Ethereum clients [56]. This cluster exhibits high active level, with an average of about 200 packages per sender per day, it is likely to be malicious activities.

Cluster 8 is composed by about 80-90 senders from 3 /24 subnet, which located in Washington DC, Hong Kong and Brussels. All of those senders are under domain name " googleusercontent.com", which is used by users of google. They send packets to 10 ports, 5060/TCP receivesthe most trafficand, then 3389/TCP and others

[2]https://www.bleepingcomputer.com/news/security/protecting-collocated-servers-from-ddos-attacks-using-gre-tunnels/

[3]https://krebsonsecurity.com/2016/09/krebsonsecurity-hit-with-record-ddos/

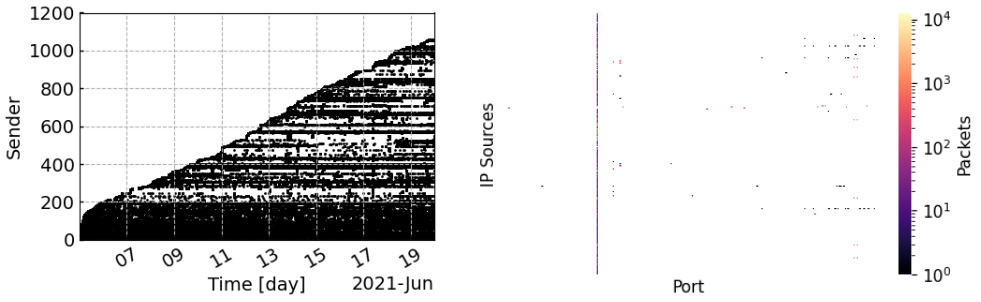


(a) General activities **(b)** Ports

Figure 8.20: Activity pattern of cluster 8.

receives same amount of traffic (see 8.20b). From 8.20a it is evident that their activities are highly coordinated.

Cluster 98 first emerged on June 10 and remained survived until the end of the monitoring period. The cluster is quite unstable, its size progressively expanded in subsequent days and its silhouette value fluctuated around 0.5. Over its 10-day lifespan, it attracted more than 1,000 IP addresses, with over 700 present for only a single day. These IP addresses are distributed across over 30 countries. Nearly 90% of its traffic is directed towards port 22/TCP. Figure 8.21 displays its activity pattern. The majority of these IP addresses had not been previously observed in the darknet before June 9 and quickly returned to inactivity. This cluster is potentially involved in malware spreading.



(a) General activities **(b)** Ports

Figure 8.21: Activity pattern of cluster 98.

8.4 Summary

Regarding the clustering results, HDBSCAN operates as anticipated, categorizing a significant number of IP addresses randomly exposed to the darknet as Noise. Also most of the extracted clusters exhibit commendable silhouette values. For the labeled senders, the clustering outcomes align well with their respective classes. In fact, clusters with high silhouette values generally demonstrate a high level of purity.

While using conventional metrics like ARI and NMI to track changes in clustering, they offer limited insights into the degree of clustering similarity across different time periods. However, these metrics fall short in providing an intuitive understanding of the evolution of clusters.

Utilizing MONIC to trace clustering changes can provide insights into the comprehensive dynamics of clusters, as well as detailed observations of activity changes within individual clusters. The findings indicate that the majority of clusters persisted daily, with new clusters emerging each day, and certain clusters transitioning to inactivity.

Understanding the types of transitions within clusters enables tracking their activities over time. Notably, very few labeled clusters emerge as time progresses, while the majority persist for long periods. In contrast, numerous unknown clusters emerged everyday, with most not persisting for long durations. Given this trend, longevity could serve as a criterion for examining unknown clusters. Manual inspection can yield valuable insights, such as expanding the understanding of sources observed in the darknet and comprehending ongoing events.

Chapter 9

Conclusion

This thesis aims to enhance the current solution for automatically analyzing darknet traffic from two perspectives, one is assessing the feasibility of adopting federated learning to enable collaboration among different data sources and obtain a more comprehensive overview of activities in the darknet, the second involves estimating changes in darknet activities by monitoring the evolution of clusters over time.

Firstly I developed a federated learning solution to allow collaboration between different darknets, train a global model to create representations for IP addresses active in darknets without centralizing data. I test the proposed method on the real-world dataset contains darknet traffic collected from different continents of the world, and simulate different scenarios to stress the performance. According to the result of the leave-one-out k-NN classifier validation, in simple scenario of two darknets of equal size (/24) cooperating, the weighted F1-score increase from 0.83 of local training to 0.88, macro F1-score increase from 0.69 to 0.72, close to the case of centralizing data together. In more complex scenario, like collaboratively training with a participant who cannot observe enough traffic, the one with more traffic can still maintain the quality of its embeddings. The results indicates that the proposed federated learning approach can mitigate the drawbacks of naive approaches without compromising performance in most scenarios, it solves the problem of difficulty in sharing information in large volume data, and also fully scalable. The only limitation is the current federated learning solution requires at least one participants able to create high quality embedding, which is not a problem in naive approach that need centralize data from multiple source.

Furthermore, a novel approach was proposed to monitor and track changes in clusters. The goal is to assess changes in all clusters and provide detailed information about those changes. In comparison to the original MONIC metrics, the adjusted version aims to improve the comparison of two clustering results that include noise and do not necessarily have the exact same samples. This adjustment is designed to align with darknet traffic scenarios and provide a more

detailed reflection of activity changes, focusing on different types of transitions. The results of monitoring clustering over 20 consecutive days demonstrate that, in contrast to traditional metrics that provide only generalized results about the level of change, the MONIC methods can capture changes in the entire clustering structure and also describe changes in individual clusters. The results offer the possibility of automatically tracking a cluster, interpreting activities carried within the associated IP addresses, and obtaining more information. This enables gaining further insights into darknet activities by analyzing the evolution of clusters, with use cases including source identification and understanding ongoing incidents like attacks.

9.1 Future work

In the future, further improvements can be made in both directions.

In the federated learning work, the solution in this thesis achieves decent result only when one darknet can observe enough traffic, an enhancement can be to improve the quality of global embeddings when only low volume traffic is available in individual participants, and also it is necessary to improve its ability to handle unblance issu. Federated learning is usually used in distributed scenarios and communication cost is an important factor, which is beyond the scope of this thesis. So another improvement is to reduce transmission costs while maintain the quality of embeddings. Furthermore, we can also investigate how to preserve privacy of participants while sharing the model.

For the unsupervised learning part, the method proposed in this thesis is to perform clustering independently on each time slot, the randomness of clustering algorithm may introduce some artifacts to interfere with the inspection of the clusters, thus future studies are suggested to investigate evolutionary clustering to reduce the noise caused by clustering algorithms. And interpretation of clustering relies on manual inspection, a possible future work is to further characterize clusters and apply anomaly detection to automatically highlight clusters with high potentiality to malicious behavior.

Bibliography

[1] Luca Gioacchini, Luca Vassio, Marco Mellia, Idilio Drago, Zied Ben Houidi, and Dario Rossi. «DarkVec: automatic analysis of darknet traffic with word embeddings». In: *Proceedings of the 17th International Conference on emerging Networking EXperiments and Technologies*. ACM, Dec. 2021, pp. 76–89 (cit. on pp. 1, 6, 8).

[2] Francesca Soro, Idilio Drago, Martino Trevisan, Marco Mellia, Joao Ceron, and José J Santanna. «Are darknets all the same? On darknet visibility for security monitoring». In: *2019 IEEE international symposium on local and metropolitan area networks (LANMAN)*. IEEE. 2019, pp. 1–6 (cit. on p. 2).

[3] Barry Irwin. «A baseline study of potentially malicious activity across five network telescopes». In: *2013 5th International Conference on Cyber Conflict (CYCON 2013)*. IEEE. 2013, pp. 1–17 (cit. on p. 2).

[4] Manos Antonakakis et al. «Understanding the mirai botnet». In: *26th {USENIX} security symposium ({USENIX} Security 17)*. 2017, pp. 1093–1110 (cit. on pp. 2, 5, 16, 17, 42).

[5] Ruoming Pang, Vinod Yegneswaran, Paul Barford, Vern Paxson, and Larry Peterson. «Characteristics of internet background radiation». In: *Proceedings of the 4th ACM SIGCOMM conference on Internet measurement*. 2004, pp. 27–40 (cit. on p. 5).

[6] Michael Bailey, Evan Cooke, Farnam Jahanian, Jose Nazario, David Watson, et al. «The internet motion sensor-a distributed blackhole monitoring system.» In: *NDSS*. 2005 (cit. on p. 5).

[7] Michael Bailey, Evan Cooke, Farnam Jahanian, Andrew Myrick, and Sushant Sinha. «Practical darknet measurement». In: *2006 40th Annual Conference on Information Sciences and Systems*. IEEE. 2006, pp. 1496–1501 (cit. on p. 5).

[8] Alberto Dainotti, Karyn Benson, Alistair King, KC Claffy, Michael Kallitsis, Eduard Glatz, and Xenofontas Dimitropoulos. «Estimating internet address space usage through passive measurements». In: *ACM SIGCOMM Computer Communication Review* 44.1 (2013), pp. 42–49 (cit. on p. 5).

[9] Alberto Dainotti, Roman Amman, Emile Aben, and Kimberly C Claffy. «Extracting benefit from harm: using malware pollution to analyze the impact of political and geophysical events on the Internet». In: *ACM SIGCOMM Computer Communication Review* 42.1 (2012), pp. 31–39 (cit. on p. 5).

[10] Chansu Han, Jun'ichi Takeuchi, Takeshi Takahashi, and Daisuke Inoue. «Dark-TRACER: Early detection framework for malware activity based on anomalous spatiotemporal patterns». In: *IEEE Access* 10 (2022), pp. 13038–13058 (cit. on p. 5).

[11] Akira Tanaka, Chansu Han, Takeshi Takahashi, and Katsuki Fujisawa. «Internet-wide scanner fingerprint identifier based on TCP/IP header». In: *2021 Sixth International Conference on Fog and Mobile Edge Computing (FMEC)*. IEEE. 2021, pp. 1–6 (cit. on p. 5).

[12] Morteza Safaei Pour, Antonio Mangino, Kurt Friday, Matthias Rathbun, Elias Bou-Harb, Farkhund Iqbal, Sagar Samtani, Jorge Crichigno, and Nasir Ghani. «On data-driven curation, learning, and analysis for inferring evolving internet-of-Things (IoT) botnets in the wild». In: *Computers & Security* 91 (2020), p. 101707 (cit. on p. 6).

[13] Dvir Cohen, Yisroel Mirsky, Manuel Kamp, Tobias Martin, Yuval Elovici, Rami Puzis, and Asaf Shabtai. «DANTE: A framework for mining and monitoring darknet traffic». In: *Computer Security–ESORICS 2020: 25th European Symposium on Research in Computer Security, ESORICS 2020, Guildford, UK, September 14–18, 2020, Proceedings, Part I 25*. Springer. 2020, pp. 88–109 (cit. on p. 6).

[14] Luca Gioacchini, Luca Vassio, Marco Mellia, Idilio Drago, Zied Ben Houidi, and Dario Rossi. «i-DarkVec: Incremental Embeddings for Darknet Traffic Analysis». en. In: (), p. 26 (cit. on pp. 6, 9).

[15] Brendan McMahan, Eider Moore, Daniel Ramage, Seth Hampson, and Blaise Aguera y Arcas. «Communication-efficient learning of deep networks from decentralized data». In: *Artificial intelligence and statistics*. PMLR. 2017, pp. 1273–1282 (cit. on pp. 6, 26).

[16] Amirhossein Reisizadeh, Aryan Mokhtari, Hamed Hassani, Ali Jadbabaie, and Ramtin Pedarsani. «Fedpaq: A communication-efficient federated learning method with periodic averaging and quantization». In: *International Conference on Artificial Intelligence and Statistics*. PMLR. 2020, pp. 2021–2031 (cit. on p. 7).

[17] Jinhyun So, Basak Guler, and Salman Avestimehr. «A scalable approach for privacy-preserving collaborative machine learning». In: *Advances in Neural Information Processing Systems* 33 (2020), pp. 8054–8066 (cit. on p. 7).

[18] Jonas Geiping, Hartmut Bauermeister, Hannah Dröge, and Michael Moeller. «Inverting gradients-how easy is it to break privacy in federated learning?» In: *Advances in Neural Information Processing Systems* 33 (2020), pp. 16937–16947 (cit. on p. 7).

[19] Tian Li, Anit Kumar Sahu, Manzil Zaheer, Maziar Sanjabi, Ameet Talwalkar, and Virginia Smith. «Federated optimization in heterogeneous networks». In: *Proceedings of Machine learning and systems* 2 (2020), pp. 429–450 (cit. on p. 7).

[20] Qiang Yang, Yang Liu, Tianjian Chen, and Yongxin Tong. «Federated machine learning: Concept and applications». In: *ACM Transactions on Intelligent Systems and Technology (TIST)* 10.2 (2019), pp. 1–19 (cit. on p. 7).

[21] Cosmin I Bercea, Benedikt Wiestler, Daniel Rueckert, and Shadi Albarqouni. «Federated disentangled representation learning for unsupervised brain anomaly detection». In: *Nature Machine Intelligence* 4.8 (2022), pp. 685–695 (cit. on p. 7).

[22] Daniel Garcia Bernal, Lodovico Giaretta, Sarunas Girdzijauskas, and Magnus Sahlgren. «Federated word2vec: Leveraging federated learning to encourage collaborative representation learning». In: *arXiv preprint arXiv:2105.00831* (2021) (cit. on p. 7).

[23] Yiqiang Chen, Xin Qin, Jindong Wang, Chaohui Yu, and Wen Gao. «Fedhealth: A federated transfer learning framework for wearable healthcare». In: *IEEE Intelligent Systems* 35.4 (2020), pp. 83–93 (cit. on p. 7).

[24] Dashan Gao, Ce Ju, Xiguang Wei, Yang Liu, Tianjian Chen, and Qiang Yang. «Hhhfl: Hierarchical heterogeneous horizontal federated learning for electroencephalography». In: *arXiv preprint arXiv:1909.05784* (2019) (cit. on p. 7).

[25] Zhuo Chen, Na Lv, Pengfei Liu, Yu Fang, Kun Chen, and Wu Pan. «Intrusion detection for wireless edge networks based on federated learning». In: *IEEE Access* 8 (2020), pp. 217463–217472 (cit. on p. 7).

[26] Hossein Fereidooni, Alexandra Dmitrienko, Phillip Rieger, Markus Miettinen, Ahmad-Reza Sadeghi, and Felix Madlener. «Fedcri: Federated mobile cyber-risk intelligence». In: *Network and Distributed Systems Security (NDSS) Symposium*. 2022 (cit. on p. 7).

[27] Talha Ongun, Simona Boboila, Alina Oprea, Tina Eliassi-Rad, Jason Hiser, and Jack Davidson. «CELEST: Federated Learning for Globally Coordinated Threat Detection». In: *arXiv preprint arXiv:2205.11459* (2022) (cit. on p. 7).

[28] Félix Iglesias and Tanja Zseby. «Pattern discovery in internet background radiation». In: *IEEE Transactions on Big Data* 5.4 (2017), pp. 467–480 (cit. on p. 8).

[29] Michalis Kallitsis, Rupesh Prajapati, Vasant Honavar, Dinghao Wu, and John Yen. «Detecting and Interpreting Changes in Scanning Behavior in Large Network Telescopes». In: *IEEE Transactions on Information Forensics and Security* 17 (2022), pp. 3611–3625 (cit. on p. 8).

[30] Francesca Soro, Mauro Allegretta, Marco Mellia, Idilio Drago, and Leandro M Bertholdo. «Sensing the noise: Uncovering communities in darknet traffic». In: *2020 Mediterranean Communication and Computer Networking Conference (MedComNet)*. IEEE. 2020, pp. 1–8 (cit. on p. 8).

[31] Vincent D Blondel, Jean-Loup Guillaume, Renaud Lambiotte, and Etienne Lefebvre. «Fast unfolding of communities in large networks». In: *Journal of statistical mechanics: theory and experiment* 2008.10 (2008), P10008 (cit. on p. 8).

[32] Danilo Giordano, Stefano Traverso, Luigi Grimaudo, Marco Mellia, Elena Baralis, Alok Tongaonkar, and Sabyasachi Saha. «YouLighter: A cognitive approach to unveil YouTube CDN and changes». In: *IEEE Transactions on Cognitive Communications and Networking* 1.2 (2015), pp. 161–174 (cit. on p. 8).

[33] Andrea Morichetta and Marco Mellia. «LENTA: Longitudinal exploration for network traffic analysis from passive data». In: *IEEE Transactions on Network and Service Management* 16.3 (2019), pp. 814–827 (cit. on p. 8).

[34] Radim Řehůřek and Petr Sojka. «Software Framework for Topic Modelling with Large Corpora». English. In: *Proceedings of the LREC 2010 Workshop on New Challenges for NLP Frameworks*. `http://is.muni.cz/publication/884893/en`. Valletta, Malta: ELRA, May 2010, pp. 45–50 (cit. on p. 9).

[35] Ricardo JGB Campello, Davoud Moulavi, and Jörg Sander. «Density-based clustering based on hierarchical density estimates». In: *Advances in Knowledge Discovery and Data Mining: 17th Pacific-Asia Conference, PAKDD 2013, Gold Coast, Australia, April 14-17, 2013, Proceedings, Part II 17*. Springer. 2013, pp. 160–172 (cit. on p. 12).

[36] Leland McInnes, John Healy, and Steve Astels. «hdbscan: Hierarchical density based clustering.» In: *J. Open Source Softw.* 2.11 (2017), p. 205 (cit. on p. 12).

[37] Martin Ester, Hans-Peter Kriegel, Jörg Sander, Xiaowei Xu, et al. «A density-based algorithm for discovering clusters in large spatial databases with noise.» In: *kdd*. Vol. 96. 34. 1996, pp. 226–231 (cit. on p. 12).

[38] *The Best IP Geolocation Database | IPIP.NET*. `https://en.ipip.net/` (cit. on p. 17).

[39] *Net systems research*. `http://www.netsystemsresearch.com/` (cit. on p. 17).

[40] *Exposure Management and Threat Hunting Solutions | Censys*. `https://censys.io/` (cit. on p. 17).

[41] *Shodan Search Engine*. `https://www.shodan.io/` (cit. on p. 17).

[42] *Internet Census Group*. `https://www.internet-census.org/` (cit. on p. 17).

[43] *The Shadowserver Foundation*. `https://www.shadowserver.org/` (cit. on p. 17).

[44] *Rapid7 - Practitioner-First Cybersecurity Solutions*. `https://www.rapid7.com/` (cit. on pp. 17, 20).

[45] *UMich ECE Research Scans*. `https://ece-scan.eecs.umich.edu/` (cit. on p. 17).

[46] *ONYPHE | Attack Surface Management Cyber Defense Search Engine*. `https://www.onyphe.io/` (cit. on p. 17).

[47] *BinaryEdge*. `https://www.binaryedge.io/` (cit. on p. 17).

[48] *Ark - CAIDA*. `https://www.caida.org/projects/ark/` (cit. on p. 17).

[49] *Stretchoid*. `https://stretchoid.com/` (cit. on p. 17).

[50] *SecurityTrails: Data Security, Threat Hunting, and Attack Surface Management*. `https://securitytrails.com/` (cit. on p. 17).

[51] *cyber.casa*. `https://www.cyber.casa/` (cit. on p. 17).

[52] *GeoLite2 Free Geolocation Data*. `https://dev.maxmind.com/geoip/geolite2-free-geolocation-data` (cit. on p. 18).

[53] *tldextract*. `https://github.com/john-kurkowski/tldextract` (cit. on p. 20).

[54] *Public Suffix List*. `https://publicsuffix.org/` (cit. on p. 20).

[55] Myra Spiliopoulou, Irene Ntoutsi, Yannis Theodoridis, and Rene Schult. «Monic: modeling and monitoring cluster transitions». In: *Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining*. 2006, pp. 706–711 (cit. on p. 38).

[56] Kazuki Hara, Teppei Sato, Mitsuyoshi Imamura, and Kazumasa Omote. «Profiling of malicious users targeting ethereum's rpc port using simple honeypots». In: *2020 IEEE International Conference on Blockchain (Blockchain)*. IEEE. 2020, pp. 1–8 (cit. on p. 53).