Universidad
Carlos III de Madrid

TESIS DOCTORAL

EXPERIMENTAL ANALYSIS OF THE SOCIO-ECONOMIC PHENOMENA IN THE BITTORRENT ECOSYSTEM

Autor:
Michał Kryczka

Directores:
Dr. Arturo Azcorra
Dr. Rubén Cuevas

DEPARTAMENTO DE INGENIERÍA TELEMÁTICA

Leganés, Diciembre 2012

Universidad
Carlos III de Madrid

Ph.D. THESIS

EXPERIMENTAL ANALYSIS OF THE SOCIO-ECONOMIC PHENOMENA IN THE BITTORRENT ECOSYSTEM

Author:
Michał Kryczka

Directors:
Dr. Arturo Azcorra
Dr. Rubén Cuevas

DEPARTMENT OF TELEMATIC ENGINEERING

Leganés, December 2012

Acknowledgements

First and foremost I wish to express my sincere thanks to the directors of my thesis. I thank Arturo Azcorra for giving me an opportunity to work for Institute IMDEA Networks and for guiding me through my years in Madrid. I am extremely grateful to Rubén, not only my supervisor but primarily my friend, for the continuous support of my Ph.D. study, tremendous source of guidance and for his motivation, invaluable feedback and patience to me.

Besides, it is not possible to mention everyone, so I want to thank collectively all the friends and colleagues from IMDEA Networks and the University Carlos III. Thanks to them, my stay in Madrid was an unforgettable experience.

Wreszcie chciałem jak najgoręcej podziękować moim rodzicom za ich cierpliwość oraz wsparcie okazane mi przez te wszystkie lata. To im dedykuję tę pracę.

Resumen

BitTorrent es la aplicación peer-to-peer para compartición de ficheros de mayor éxito y responsable de una fracción importante del tráfico de Internet. Trabajos previos han estudiado BitTorrent usando técnicas de simulación, modelos analíticos y medidas reales. Aunque las técnicas analíticas y de simulación son más sencillas de aplicar, típicamente presentan versiones simplificadas de los problemas analizados y en el caso concreto de BitTorrent pueden obviar aspectos o interacciones fundamentales que ocurren en los *enjambres* de BitTorrent. Por lo tanto, en esta tesis utilizaremos como pilar de nuestra investigación técnicas de medidas reales. En primer lugar presentaremos un resumen de las técnicas de medidas usadas hasta el momento en el ámbito de BitTorrent que suponen la base teórica para el diseño de nuestras propias herramientas de medida que nos permitirán analizar enjambres reales de BitTorrent. Usando los datos obtenidos con estas herramientas estudiaremos aspectos diferentes de BitTorrent con un enfoque especial de los aspectos socioeconómicos. En la primera parte de la tesis, realizaremos un estudio detallado de la topología de los enjambres reales de BitTorrent así como de detalles acerca de las interacciones entre peers. Nuestro análisis demuestra que la resistencia de la topología de los enjambres reales de BitTorrent es menor que la ofrecida por grafos aleatorios equivalentes. Además, los resultados revelan que las políticas de los Provedores de Internet junto con la incipiente utilización de clientes de BitTorrent modificados y otros efectos en la red (p.ej. congestión) hacen que los enjambres reales de BitTorrent presentan una composición de *localidad*. Es decir, un nodo tiene un número de vecinos dentro de su mismo Proveedor de Internet mayor del que obtendría en una topología puramente aleatoria. Estos resultados son de interés para las empresas que utilizan BitTorrent en sus operaciones, así como para los Provedores de Internet responsables de transportar el tráfico de BitTorrent. En la segunda parte de la tesis, analizamos los aspectos de publicación de contenido en los mayores portales de BitTorrent. En concreto, los resultados presentados muestran que sólo un pequeño grupo de publicadores (alrededor de 100) es responsable de hacer

disponible más de dos tercios del contenido publicado. Además estos publicadores se pueden dividir en dos grupos: *(i)* aquellos con incentivos económicos y *(ii)* publicadores de contenido falso. El primer grupo hace disponible contenido protegido por derechos de autor (que es típicamente muy popular) en los principales portales de BitTorrent con el objetivo de atraer a los consumidores de dicho contenido a sus propios sitios web y obtener un beneficio económico. La eliminación de este grupo puede tener un impacto importante en la popularidad de los principales portales de BitTorrent así como en el tráfico generado por BitTorrent en Internet. El segundo grupo es responsable de la publicación de contenidos falsos. La mayor parte de dichos contenidos están asociados a una actividad maliciosa (p.ej. la distribución de software malicioso) y por tanto suponen una seria amenaza para el ecosistema de BitTorrent, en particular, y para Internet en general. Para minimizar los efectos de la amenaza que presentan estos publicadores, en la última parte de la tesis presentaremos una nueva herramienta denominada TorrentGuard para la pronta detección de contenidos falsos. Esta herramienta puede accederse a través de un portal web y a través de un plugin del cliente de BitTorrent Vuze. Finalmente, presentamos MYPROBE, un portal web que permite consultar una base de datos con información actualizada sobre los publicadores de contenidos en BitTorrent.

Abstract

BitTorrent is the most successful Peer-to-Peer (P2P) application and is responsible for a major portion of Internet traffic. It has been largely studied using simulations, models and real measurements. Although simulations and modelling are easier to perform, they typically simplify analysed problems and in case of BitTorrent they are likely to miss some of the effects which occur in real swarms. Thus, in this thesis we rely on real measurements. In the first part of the thesis we present the summary of measurement techniques used so far and we use it as a base to design our tools that allow us to perform different types of analysis at different resolution level. Using these tools we collect several large-scale datasets to study different aspects of BitTorrent with a special focus on socio-economic aspects. Using our datasets, we first investigate the topology of real BitTorrent swarms and how the traffic is actually exchanged among peers. Our analysis shows that the resilience of BitTorrent swarms is lower than corresponding random graphs. We also observe that ISP policies, locality-aware clients and network events (e.g., network congestion) lead to locality-biased composition of neighbourhood in the swarms. This means that the peer contains more neighbours from local provider than expected from purely random neighbours selection process. Those results are of interest to the companies which use BitTorrent for daily operations as well as for ISPs which carry BitTorrent traffic. In the next part of the thesis we look at the BitTorrent from the perspective of the content and content publishers in a major BitTorrent portals. We focus on the factors that seem to drive the popularity of the BitTorrent and, as a result, could affect its associated traffic in the Internet. We show that a small fraction of publishers (around 100 users) is responsible for more than two-thirds of the published content. Those publishers can be divided into two groups: *(i)* profit driven and *(ii)* fake publishers. The former group leverages the published copyrighted content (typically very popular) on BitTorrent portals to attract content consumers to their web sites for financial gain. Removing this group may have a significant impact on the popularity of BitTorrent portals and, as a result, may affect a big portion of the Inter-

net traffic associated to BitTorrent. The latter group is responsible for fake content, which is mostly linked to malicious activity and creates a serious threat for the BitTorrent ecosystem and for the Internet in general. To mitigate this threat, in the last part of the thesis we present a new tool named TorrentGuard for the early detection of fake content that could help to significantly reduce the number of computer infections and scams suffered by BitTorrent users. This tool is available through web portal and as a plugin to Vuze, a popular BitTorrent client. Finally, we present MYPROBE, the web portal that allows to query our database and to gather different pieces of information regarding BitTorrent content publishers.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

BitTorrent standard [44] was first published by Brian Cohen in 2001. Since that time, the protocol has become the most successful Peer-to-Peer (P2P) file-sharing application. It is currently used by hundreds of millions of users and is responsible for a large portion of Internet traffic [23]. BitTorrent is an alternative to traditional server-client content distribution. It is indeed very effective in distributing large files, e.g., movies, big updates or open-source software distributions, and is used by some companies to perform important tasks such as software release or content replication. This effectiveness has led to a big success of the protocol among the users and has created a huge ecosystem that contains millions of BitTorrent peers, several BitTorrent client implementations, different trackers and torrent-discovery sites. It is estimated that nowadays the number of monthly active users can reach a quarter of a billion [1].

Because of the popularity of BitTorrent and its wide presence in the Internet, it is very important to obtain a full knowledge about technical aspects and implications of the protocol as well as about all the mechanisms that drive BitTorrent popularity. Nowadays, BitTorrent has wide commercial appliance. It is used to distribute the software, both by the companies (for example by Blizzard to distribute their games) as well as by many free software projects. BitTorrent can be also used by the companies to share the content across their data centres, for example Facebook [2] and Twitter [3] use BitTorrent for distributing updates to their servers. Thus, it is important to investigate such information as the efficiency of a swarm to disseminate content, the resilience of a swarm to different events such as being partitioned or the efficiency and overhead generated by swarming and neighbour selection algorithms. Moreover, because of the fact that BitTorrent is responsible for a significant part of the Internet traffic overall, it is of interest to Internet Service Providers which carry

this traffic. Those providers are interested to understand how different events caused by BitTorrent affects their business (for example locality in the neighbourhood of BitTorrent clients). Furthermore, there are not only technical factors that are important to fully understand the BitTorrent mechanisms. The extreme popularity of P2P applications (and especially BitTorrent) that generates their associated traffic and thus their impact on the network, seems to be primarily affected by social and economic factors. More specifically, the big popularity of BitTorrent is because it is widely adopted to distribute the copyrighted files (e.g., recent release of Hollywood movies) at no cost to the downloaders. There are several torrent portals (e.g., the Pirate Bay) that index millions of content, which in majority violates copyright. Publishing this content is not only unlawful and can have serious legal implications but it also requires a significant amount of resources. Thus, it seems important to analyse the socio-economic factors of content publishing phenomena in BitTorrent. Especially, it is crucial to understand how the business that drives BitTorrent popularity looks like and to reveal who is responsible for publishing the content (and committing copyright infringement) as well as what are their incentives and possible benefits.

The popularity of BitTorrent attracted an interest in the research community. In recent years, a significant progress has been made in order to understand the strengths and limitations of BitTorrent's protocol and its tit-for-tat mechanism [45, 60, 67, 69, 71, 78, 80, 81, 87, 89, 96, 108]. The research community has also examined various aspects of swarming mechanism in BitTorrent [39, 62, 73, 80, 91]. Moreover, there are several works which analyse performance aspects and propose different techniques that can improve BitTorrent functionality [34, 35, 43, 52, 57, 78, 92, 94, 96, 100, 110]. Furthermore, other aspects of BitTorrent such as demography of users [67, 99, 118] along with incentives [31, 68, 82, 103] issues were investigated. There are also some works on security [54–56], especially on analysis of the vulnerabilities of BitTorrent protocol to free-riders [86, 89, 108] and on privacy issues [38, 42]. Schemes to enhance BitTorrent to support streaming applications [40, 53, 102, 112] were also proposed.

However, despite its importance, the analysis of BitTorrent socio-economic aspects has received little attention. Note that these socio-economic components are important factors, which in fact are the reasons of BitTorrent success and they are crucial for understanding the popularity of BitTorrent. In the literature, we can find just few previous works [41, 50, 115] that address some specific economic aspects of BitTorrent such as techniques to reduce the impact of BitTorrent in the transit traffic of ISPs, which lowers their operational costs. In this thesis we try to analyse several

relevant socio-economic aspects of the BitTorrent ecosystem. In order to understand them, we propose measurement techniques and perform extensive data collection processes. There are several proposals on how to simulate and model the BitTorrent ecosystem [62, 100, 116]. However, simulating and modelling the BitTorrent ecosystem have several limitations as they may miss some of the effects (like specific network events) that occur in real BitTorrent swarms. They also do not allow to gather some specific data like downloaders' demographic or publishers' behaviour. Thus, in order to get the most accurate and complete results, we use real data from crawling BitTorrent in real conditions.

The first objective of this thesis is to present an extensive analysis of current BitTorrent measurement techniques [75]. Based on it, we implemented a measurement tool which is able to monitor the BitTorrent ecosystem and obtain various types of data. The level of details of crawled data can be adjusted in function of scalability we want to obtain. We performed several different crawlings between 2008 and 2012. The majority of our measurements are based on the Pirate Bay portal [21], which is a publicly accessible portal, contrary to private portals called BitTorrent darknets [88, 117]. According to the Alexa Rank [9] and based on earlier studies [118], the Pirate Bay is the most popular and representative BitTorrent portal. Maximum number of their daily visits, as reported by Alexa, reveals that the Pirate Bay is the most visited portal and it receives at least twice more visits than the second largest portal, called Torrentz. In Appendix A we discuss the representativeness of the obtained results beyond popular BitTorrent portals. Using the obtained dataset we perform an exhaustive analysis of BitTorrent and we aim to provide a comprehensive picture of the BitTorrent ecosystem focusing on different socio-economic aspects.

To understand socio-economic factors which affect BitTorrent, we use different datasets collected with our measurement tools. We address socio-economic issues by focusing on two main characteristics: *(i)* connectivity properties of BitTorrent swarms and *(ii)* behaviour and incentives of content publishers. The analysis of the connectivity properties in real BitTorrent swarms can reveal us important information like locality-bias exhibited by current swarms, the resilience of a swarm to different events or efficiency of a swarm to disseminate content. Unveiling who publishes content in major BitTorrent portals and why, allows us to investigate the main incentives of major content publishers. Especially, we dedicate our effort to understand and mitigate fake publishers who perform a continuous content poisoning attack against major BitTorrent portals that affects millions of downloaders.

We start with addressing the fundamental questions regarding the connectivity

properties of real BitTorrent swarms [77]. For this purpose, we collected the evolution of the graph topology of real torrents to carefully study the overlay topology characteristics of real BitTorrent swarms as well as the connectivity properties of the peers. We dedicate special effort to the locality effect. BitTorrent traffic creates a high amount of transit traffic which goes to transit links. This consequently increases the operational costs of the ISPs. Several papers proposed solutions to address this problem. However, in this thesis we are the first to investigate whether or not locality already takes place in the real world. Based on our results, we demonstrate that current BitTorrent swarms are experiencing a marked locality phenomenon at the overlay construction level (or connectivity graph). This locality effect is even more pronounced when we consider the exchange traffic relationships between peers. This suggests that an important portion of the BitTorrent traffic is currently confined within the ISPs. We also focus on the structural characteristics of BitTorrent swarms. This is an important problem as some critical functions of the content distribution architecture of some companies depend on BitTorrent. For example, BitTorrent is often used for the distribution of the content (e.g., Linux distributions) or by big companies to share the content across their data centres. Our results demonstrate that, contrary to the conclusion of previous studies, the overlay topologies of real swarms cannot be modelled as random graphs. Furthermore, the main characteristics of these swarms (clustering coefficient and characteristic path length) suggest that they present a relatively efficient topology to disseminate information but they are significantly less resilient to attacks than random graphs. The analysis of the peer level connectivity properties reveals that peers continuously change more than half of their neighbours, which may generate an unnecessary communication overhead. Furthermore, a leecher typically keeps stable connections with a handful of neighbours with which it exchanges most of its traffic, whereas seeders do not establish long-term connections with any peer in order to guarantee the homogeneous distribution of pieces among the leechers.

To enhance the picture of the BitTorrent economic model, next we focus our effort to understand the content publishing phenomena in BitTorrent [48, 49]. The growing popularity of BitTorrent is primarily due to the availability of valuable content without any cost for consumers. However, besides the required resources, publishing valuable (and often copyrighted) content has serious legal implications for the users who publish the material. This raises the question that whether (at least major) content publishers behave in an altruistic fashion or have other motives such as financial incentives. We conduct a systematic study on the major BitTorrent publishers and we discover that a small fraction of publishers (3%) is responsible for

publishing a majority amount of the content (two-thirds) that serve three-quarters of the downloads. Our investigation reveals several key insights. First, anti-piracy agencies and malicious users publish a large amount of "fake" files to protect copyrighted content and spread malware or scam downloaders, respectively. Fake content represents an important portion of those files shared in BitTorrent and overwhelming majority of the analysed fake files is linked to either malware or scam websites. This creates a serious threat for the BitTorrent ecosystem which should be mitigated. Second, excluding the fake publishers, content publishing in major BitTorrent portals appears to be largely driven by companies that try to attract consumers to their own web sites for financial gain. Therefore, if these companies lose their interest or are unable to publish content, the popularity of the BitTorrent portals and their associated traffic may significantly decrease. We also demonstrate that profit-driven publishers attract more loyal consumers whereas top altruistic publishers have a larger fraction of loyal consumers with a higher degree of loyalty.

Finally, we present our BitTorrent web portal [11]. We have implemented a crawler that continuously monitors the BitTorrent ecosystem. The portal offers an interface to the gathered data and it makes possible to query our database in order to obtain information about content publishers. It allows to monitor the activity of top publishers of BitTorrent and checks such details like number of published torrents, their category or ISP used by the publishers. What is more important, it also implements a module (called TorrentGuard [26, 76]) which monitors published content and detects if the content is fake. The module continuously checks the IP address of the initial seeders of each torrent and compares it with the addresses used by fake publishers. TorrentGuard is also implemented in a form of a Vuze plugin. The usage of TorrentGuard could possibly reduce the number of computer infections and scams suffered by BitTorrent users.

In short the main contributions of this thesis are:

- an extensive analysis of different measurement techniques and development of a crawling software which can gather data about the BitTorrent ecosystem at different level of details
- understanding the connectivity properties of real BitTorrent swarms
- measurement and analysis of the BitTorrent locality effect and swarm resilience
- a comprehensive analysis of the publishing phenomena and a detailed description of different groups of publishers

- a detailed analysis of the presence of fake content in the BitTorrent ecosystem
- a portal that monitors BitTorrent and offers detailed information about publishers
- a novel solution which allows the early detection of fake content in the BitTorrent ecosystem

The rest of the thesis is structured as follows. In Section 2 we present the background of BitTorrent together with the work related to our research. Next, in Section 3 we discuss different methods of measuring BitTorrent. In Section 4 we focus on topology of BitTorrent swarms and its implication in swarming efficiency, resilience and locality, while in Section 5 we investigate the different profiles of BitTorrent publishers. Finally, in Section 6 we are presenting our web application, before concluding in Section 7.

Chapter 2

Background and Related Work

2.1 Background

BitTorrent [45] is the name used by Brian Cohen to define the peer-to-peer file sharing protocol that he designed one decade ago. The great success of this protocol led to a creation of a complex system around it. We adopt the terminology used by Zhang et al. [118] to refer to this system as the *BitTorrent Ecosystem*. In this Section we describe the main players of the ecosystem as well as its functionality. This is summarised in Fig. 2.1.

2.1.1 Main Elements of the BitTorrent Ecosystem

The main elements of the BitTorrent ecosystem are as follows:

- BitTorrent Portals: these are web pages which index .torrent files, classify them into different categories and provide basic information for each file. These portals serve as rendezvous points between content publishers and BitTorrent downloaders. The publishers upload their .torrent files to BitTorrent portal and the clients download them. The Pirate Bay [21] and IsoHunt [16] are the examples of BitTorrent portals.

- .torrent file: this is a meta-information file which includes relevant information for the BitTorrent protocol such as: (i) the content infohash, this is a unique identifier of the content in the BitTorrent ecosystem; (ii) the IP address(es) of the BitTorrent tracker(s) managing the content distribution process; (iii) the size of the content and the number of pieces forming the file.

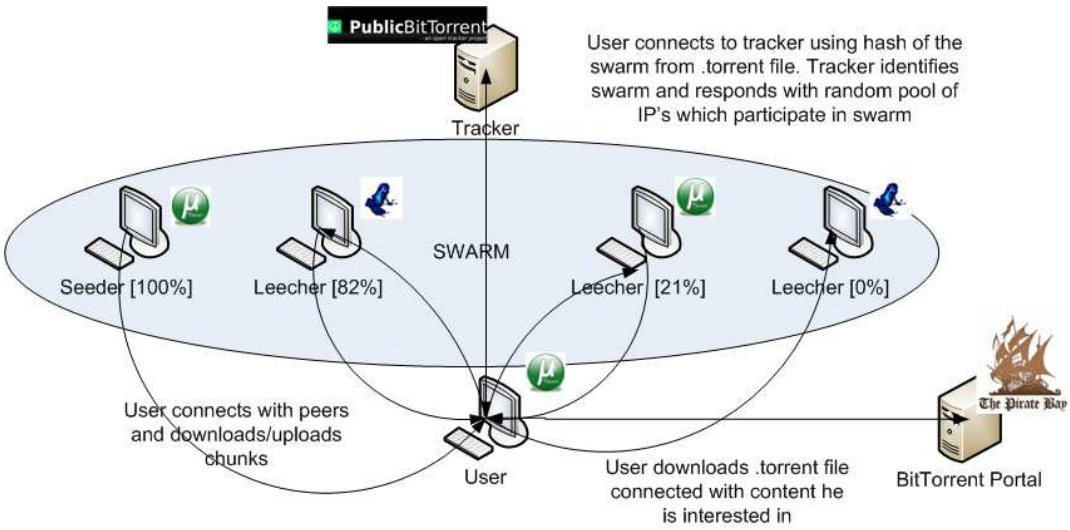


Figure 2.1: BitTorrent Ecosystem basic functionality: (i) The BitTorrent client contacts a BitTorrent portal to download the .torrent file associated to the desired content (the .torrent file includes the IP address of the tracker managing the swarm associated to the desired content); (ii) The BitTorrent client contacts the tracker that provides the IP addresses of a set of peers within the swarm; (iii) The BitTorrent client connects to these peers for downloading the content.

- magnet link: this is an URI-like link that includes the infohash of a specific content and optionally the address(es) of a tracker(s) [4]. A user can launch a download process retrieving the magnet link instead of the .torrent file from a BitTorrent portal. Then, using information from magnet link the user can obtain the .torrent file from other peers in the swarm[1]. The magnet links have recently become significantly important as the administrators of the largest BitTorrent portal, the Pirate Bay, stopped serving .torrent file since March 1st 2012. Instead, they serve exclusively magnet links [5].

- BitTorrent trackers: these are servers that manage the BitTorrent download process of a given content. The set of peers downloading a given file is named *swarm*. The tracker maintains a list with the IP addresses and the download progress of all the peers forming the swarm associated to a specific content. Furthermore, when a new peer joins the swarm, it contacts the tracker in order to obtain a list of IP addresses of other peers participating in the swarm. By

[1]The magnet link can be also used as index to retrieve the associated .torrent file from the different DHTs implemented by BitTorrent clients [47].

doing so, the new incomer is able to retrieve pieces of the content from these peers.

- BitTorrent downloaders (peers): these are clients forming the swarm that download and/or upload pieces of the content. We distinguish two types of peers. A *seeder* is a peer that possesses a complete copy of the content, thus only uploads pieces whereas a *leecher* does not have the complete file so it uploads and downloads pieces.

- BitTorrent publishers: these are the clients that make available the first copy of the content in the BitTorrent ecosystem. In this thesis we will also call them as *initial seeders*.

2.1.2 Publishing Content in BitTorrent

The files, which are to be shared with BitTorrent protocol, are divided into k pieces of equal size (typically 256 KB to 4 MB) named *chunks* that are further divided into *blocks* (typically 16 KB). The block is the data unit to be exchanged among the peers. In order to make available a content C in BitTorrent, the content publisher creates a *.torrent* file associated to C. This .torrent file includes, along with other information, the IP address(es) of the tracker(s) that manage the download process of the content.

After creating the .torrent file, the publisher uploads it to one or more BitTorrent portals. For this purpose, it typically uses an account (with a specific username) created in these portals. Furthermore, the publisher distributes the first copy of the content by acting as the initial seeder in the associated swarm. Therefore, *the content publisher can be identified by the IP address of the initial seeder distributing the content and by the username utilised to upload the content to a BitTorrent portal*.

There are a few BitTorrent portals such as the Pirate Bay[2] indexing millions of torrents and receiving millions of daily visits. These portals are critical for the BitTorrent ecosystem as demonstrated by Zhang et al. [118]. They offer detailed information regarding each indexed torrent. This information slightly varies from one portal to another, but in general it includes: category of the content, number of associated files, size of the whole content in the torrent, complete name of the file, upload date, username who uploaded the torrent, number of seeders and leechers participating in the torrent swarm (this data is updated every few minutes) and a description text giving more detailed information regarding the content. Finally, it

[2]This is the current largest BitTorrent portal based on Alexa Ranking.

is worth noting that some of these major portals offer an RSS feed to announce the newly published torrents.

We use the Pirate Bay as the reference BitTorrent portal throughout the thesis. Previous works [118] demonstrated that the Pirate Bay is a key element and the most important portal in the BitTorrent ecosystem. The Pirate Bay relies on publishers who need to create a user account in order to upload .torrent files, whereas other portals, such as IsoHunt, use crawling techniques to obtain the offered content from third portals such as the Pirate Bay. Hence, the Pirate Bay is the most interesting portal to be considered in order to understand different characteristics of the BitTorrent ecosystem including the content publishing phenomenon. Moreover, maximum number of their daily visits, as reported by Alexa [9], reveals that the Pirate Bay is the most visited portal and it receives at least twice more visits than the second largest portal, called Torrentz. This confirms that the Pirate Bay is the most representative BitTorrent portal. In Appendix A we discuss the representativeness of our obtained results beyond popular BitTorrent portals. The Pirate Bay offers the following relevant services to our study: (i) an RSS feed system [22] in which each new published content is announced along with the username that uploaded the .torrent file to the portal; (ii) each user registered within the Pirate Bay portal has an individual web page in which its published torrents are listed and (iii) the Pirate Bay removes the accounts, web pages and .torrent files of those users whose content is detected as fake. Typically, this happens after a client, who downloaded the content, reports its falseness to the Pirate Bay administrators.

2.1.3 Joining a BitTorrent Swarm and Discovering Peers

When a BitTorrent user wants to download a given content C, it looks for the .torrent file (or magnet link) associated to C in a BitTorrent portal and downloads it. The .torrent file (or magnet link) can be opened with any of the existing BitTorrent clients [14]. Upon opening the .torrent file, the BitTorrent client connects to one of the included trackers.

The tracker is a central entity that knows the IP addresses of all the peers sharing a given content, e.g., C, as well as their download progress. In practice, the BitTorrent ecosystem relies on few trackers that manage a large number (up to few millions) of torrents in parallel. The OpenBitTorrent and PublicBitTorrent trackers[3] are currently the most important ones.

The set of all the peers sharing a content is named *swarm*. In order to join a

[3] `www.openbittorrent.org` and `www.publicbt.org`

swarm, a new peer first contacts the tracker using an *announce started* request that is answered by the tracker with the number of seeders and leechers participating in the swarm along with the IP addresses of N (between 40 and 200) randomly selected peers. These N peers form the initial neighbourhood of the new node. Furthermore, if a peer's neighbourhood size falls below a given threshold (typically 20) it sends again an *announce started* request to the tracker in order to get new neighbours. Finally, when a peer leaves the swarm, it sends an *announce stopped* request to the tracker that removes this peer from the list of participants in the swarm.

2.1.4 BitTorrent Delivery Procedure

There are two types of nodes within a BitTorrent swarm. On the one hand, *seeders* are those nodes which have a full copy of the file and only upload chunks. On the other hand, *leechers* are those nodes which do not have the complete file, thus, they upload and download chunks.

In BitTorrent two peers communicate using the *peer wire* protocol. Every communication starts with an initial *handshake*. Once the handshaking sequence is completed (and before any other messages are sent) the peers exchange the *bitfields* using a BITFIELD message. The bitfield indicates which chunks of the file a peer has already downloaded. Furthermore, every time a peer gets a new chunk, it informs its neighbours using a *HAVE* message. Hence, every peer is aware of the chunks that each neighbour has at every moment.

BitTorrent uses the *Tit-for-Tat* as an incentive model for the delivery mechanism. Basically, each leecher uploads chunks to those other leechers from whom it is downloading more chunks. The *Choking Algorithm* is responsible for providing this behaviour. It is a periodical operation where every 10 seconds leecher selects (unchokes) n other leechers from its neighbourhood to upload chunks to. These n (typically 4) unchoked leechers are those from whom the peer downloaded more chunks during the last 30 seconds. The rest of the neighbours are blocked (choked). In the case of seeders, they unchoke n (typically 4) leechers to whom more chunks they uploaded in the last 30 seconds (i.e., those with higher download rate). In addition to the regular unchoke operation, BitTorrent implements the *optimistic unchoke* operation. Every 30 seconds (this is, every 3 regular unchoke operations) both leechers and seeders randomly select one choked neighbour to upload chunks to. Finally, when a leecher is unchoked by a neighbour, it applies the *Rarest First Policy* in order to choose which chunk to request to this neighbour. Since leechers have full knowledge about the availability of every chunk in its neighbourhood, it always requests

the rarest one.

2.1.5 BitTorrent Extension

Several extensions to BitTorrent have been proposed so far. Here we just mention those relevant to our measurement studies:

- Distributed Hash Table (DHT): trackers are a single point of failure in the BitTorrent ecosystem. Indeed, they are typically threatened by legal actions [6, 7]. The BitTorrent developers reacted to this by designing a tracker-less mechanism that allows a BitTorrent user learning the IP addresses of peers without contacting the tracker. This mechanism is based on a DHT [109].

- Peer Exchange (PEX): This is a simple gossiping protocol that is used to get IP addresses of peers participating in the swarm. In more detail, PEX works as follows: a given peer P sends a PEX request to one of its neighbours, e.g., N. If N supports PEX, it responds with the list of IP addresses of its neighbours. Hence, by using few PEX queries a given peer can learn the IP addresses of a large number of participants in the swarm without requesting them from the tracker. A more detailed description of the functionality of PEX can be found at [114].

2.2 Related Work

This Section covers the specific related work for the different contributions of this thesis. They are presented in a separate manner so that, the reader can have a detailed view of the overall research effort conducted in the different areas covered by the thesis. First, we present related work in the area of BitTorrent measurement techniques. Next, we focus on work related to BitTorrent connection properties, especially to locality technique, resilience of the swarm and topological characteristics of the swarm. Finally, we present work about BitTorrent publishers and we discuss the existence of fake content in the BitTorrent.

2.2.1 BitTorrent Measurement Techniques

Several authors used real data collection in order to understand different aspects of the BitTorrent [50, 62, 96]. Different methods of measuring the BitTorrent are

described in [75]. In this subsection we present the most representative literature in BitTorrent measurement techniques sorted chronologically.

M. Izal et al. [67] performed one of the first measurement studies of BitTorrent in 2004. They study a single torrent corresponding to a Red Hat Linux distribution. The authors collect two traces. The first one is the log of the tracker managing the swarm whereas the second trace is collected using a modified BitTorrent client (i.e., a crawler) participating in this swarm. This study analyses some performance characteristics such as the existence of an initial flash-crowd, session types and duration, throughput distribution, etc. In addition, the authors use their traces to show some demographics parameters (country of the peers) for the specific studied torrent.

In 2005 we can find two higher-scale studies that looks at multiple torrents instead of just one. The first study was conducted by L. Guo et al. [62] and it uses two type of traces. The first trace [33] was collected by parsing the web page of two trackers which published the information regarding the managed torrents and the peers participating in the associated swarms. The second trace was captured from inside an ISP network and it collected the download of .torrent files from the users (the authors approximate the timestamp of the .torrent file download as the peer arrival instant to the BitTorrent ecosystem). The authors use these traces to model a bunch of BitTorrent performance parameters distribution: torrent size, torrent popularity, peer arrival rate, download, session duration, flash-crowd phenomenon, etc.

The second study was performed in the same year by J. Pouwelse et al. [99]. They perform a measurement study combining: (i) a crawling of the major BitTorrent portal at that time (Suprnova), (ii) a macroscopic tracker crawling to infer the IP address of peers participating in hundred of torrents announced in Suprnova and (iii) a microscopic crawling to infer the session time as well as the download speed of thousand of peers. Moreover, the same authors made available a second dataset[4] (different that the one used in [99]). This dataset is a combination of an active measurement study of the top 2000 torrents from The Pirate Bay and a passive measurement of the top 750 torrents from The Pirate Bay using a mix of macroscopic and microscopic techniques.

In 2007, M. Piatek et al. [66] are the first using a microscopic technique to measure the BitTorrent peers' upload capacity (not upload rate). This technique is explained in Chapter 3.2.

In 2009, G. Siganos et al. designed Apollo, a high performance BitTorrent crawler [106, 107]. Apollo systematically monitors all the peers sharing the top 600 torrents of The Pirate Bay. The IP addresses of the peers are learnt by exploiting

[4] `http://multiprobe.ewi.tudelft.nl/dataset.html`

PEX and DHT BitTorrent extensions. Furthermore, Apollo implements microscopic techniques to measure both the instantaneous download rate and the number of uploaded IP packets of the peers.

Next, there are two different studies conducted in 2010: (i) LeBlond et al. [38] perform a macroscopic measurement based on the crawling of the Pirate Bay tracker. To the best of our knowledge, the authors use the most scalable technique described so far that is able to crawl the participants of 750K torrents in around 30 min. from a single computer. The objective of this study is to demonstrate that the privacy offered by BitTorrent to the users is extremely weak. (ii) C. Zhang et al. [118] perform the most complete study of the BitTorrent ecosystem demographics so far by combining BitTorrent portals and trackers macroscopic measurements. They use a high-performance distributed architecture of crawlers to monitor multiple BitTorrent portals and trackers in parallel[5].

Finally, in [73], authors combine macroscopic tracker measurement with a microscopic study. As a difference to previous studies, authors monitor the complete life cycle of thousand of torrents. In order to learn new torrents they use the RSS feed tool offered by the studied BitTorrent portal (Mininova). The objective of this work is to study the file unavailability (i.e., lack of seeders) problem in BitTorrent.

In Chapter 3 we discuss in details different measurement techniques, the main challenges that needs to be faced, and possible solutions for them.

2.2.2 Locality in BitTorrent

One of the early works, which were treating the effect of locality, was the one by Karagiannis et al. [72]. The authors demonstrated a substantial overlap in the torrents downloaded by users located within a campus network. In [36] the authors were using simulations to investigate the effect of limiting the number of inter-AS connections on transit traffic load and end-user experience. The interest in locality is not only limited to BitTorrent, as in [29] the authors investigate the locality biasing in Gnutella. They introduce the notion of oracle that ISP supplies in order to offer a list of local neighbours to the user.

Substantial efforts have been dedicated to understand and implement BitTorrent locality solutions. Systems which implement locality solutions such as P4P [115] or ONO [41] have been proposed. First system, P4P [115], is an architecture which allows to control traffic between network providers and applications. It can be applied to different P2P applications like Pando or Liveswarms. On the other hand,

[5]Part of their dataset is available at `http://cis.poly.edu/?chao/bt-ecosys.html`

ONO [41] is a java plugin for Azureus/Vuze clients, which uses network views gathered, at low cost, from CDNs. The aim of the system is to reduce cross-ISP traffic. Lin et al. [85] propose ELP that is designed to keep traffic local to ISPs. Authors provide a model that gives bounds on the inter-ISP traffic and they model the effectiveness of the system using PlanetLab. Ren et al. [101] propose AS-aware peer-relay which can bring benefits for P2P VoIP systems. Locality biasing has also been applied to P2P streaming systems [65, 98].

Furthermore, some studies have performed thorough studies of the expected performance of these locality solutions [37, 50, 97]. Piatek et al. [97] discuss pitfalls for an ISP-friendly locality policy and ISPs traffic engineering constraints. Cuevas et al. [50] explore the impact of high locality. Similar work is done in [37] where the authors perform extensive experiments in a controlled environment to evaluate the impact of high locality on transit traffic and peers download completion time. Typically, these performance studies assume that current BitTorrent swarms correspond to a random graph structure. However, our analysis reveal that current BitTorrent swarms already show a locality-biased composition. Hence, the results of previous performance studies could be revisited using our conclusions.

In addition, some works [50, 64] have shown that the demographics of a torrent directly impact its inherent locality level and the theoretical capacity to localise traffic. For instance, a torrent for a local Japanese movie is expected to confine (even under a random overlay construction) most of the traffic within Japanese ISPs whereas a *blockbuster* popular movie is expected to be consumed by a large number of users across the world leading to a poor traffic locality but offering a large room for improvement using locality techniques. Furthermore, these works report the presence of unlocalisable torrents for a peer, i.e., torrents in which there are no other nodes from the same ISP, thus making locality impossible in practice for that peer.

However, to the best of the authors' knowledge, little effort has been dedicated to characterise the level of locality exhibited by current BitTorrent swarms. Otto et al. [95] addressed this issue by analysing the inter-country and inter-AS BitTorrent traffic using a dataset including traces for 500K users. Instead in our study, in Chapter 4, we analyse the locality-biased composition of 50K peers' neighbourhoods as well as their stable neighbours. Furthermore, we report those ISPs and countries in which we observe a more significant locality effect. We believe that the results in both studies are complementary.

2.2.3 Resilience of BitTorrent Swarms

To the best of the authors' knowledge the unique paper studying the resilience of BitTorrent swarms to be partitioned is [30]. The authors analyse similar scenarios to those considered in our study (Chapter 4), namely random nodes removal process and highest-degree nodes removal process, in a controlled environment for a single torrent. Rather, in our analysis we consider 400 swarms snapshots collected from 250 real BitTorrent swarms. In addition we compare the resilience of real BitTorrent swarms to that shown by equivalent random graphs.

2.2.4 BitTorrent Swarms Characteristics

Some efforts have been done in order to understand the topology of BitTorrent Swarms [30, 51, 80, 111]. Dale et al. [51] performed experimental study and analysed evolution of BitTorrent swarms in a controlled environment. Authors of [111] simulated BitTorrent overlay and looked at the connection matrix of the peers. Simulation was also used in [30] where the authors were examining the topology of the swarms and robustness of the overlay for churns. However, due to the difficulties in gathering real data, the existing works are based on simulation or emulation in controlled environment and a limited number of torrents. In contrary to this, in this thesis we rely in real data collected from a large number of torrents in order to understand the graph topology of live BitTorrent swarms.

To the best of the authors' knowledge, the unique previous study using real data is [61]. This work presents a similar measurement methodology to that described in Chapter 4 to collect multiple swarm snapshots for 35 very popular torrents. The authors use the collected data to analyse the clustering coefficient and the power-law properties of the node degree distribution of real BitTorrent swarms. In our study, we use a dataset including 250 torrents of different size and monitor the analysed torrents from their birth. Furthermore, we study, in addition to the clustering coefficient and the power-law properties, the following aspects of BitTorrent swarms topology: (i) we analyse the characteristic path length in real BitTorrent swarms, (ii) we compare the real BitTorrent swarms with equivalent random graphs, (iii) we study the influence of the swarm size in its topology, and (iv) we characterise the stability of BitTorrent swarms topologies along time.

2.2.5 Stable Relationship of BitTorrent Clients

Few works in the literature have analysed the existence of stable connections among peers in BitTorrent swarms. First, Legout et al. [80] used a controlled environment and few torrents to analyse the interaction between peers in a swarm. The authors conclude that peers with similar speed tend to establish stable relationships among them. This observation partially supports our result across hundreds of real BitTorrent swarms in which we observe that peers tend to keep stable connections (i.e., exchange traffic) with few of its neighbours. Second, Choffnes et al. [42] use traces from 10K peers to identify the existence of communities of BitTorrent users across torrents and time. Specifically, the authors reveal that BitTorrent users with similar interests tend to interact along time in multiple torrents leading to the creation of identifiable communities.

2.2.6 BitTorrent Publishers

The most relevant work is a study that examined the weakness of BitTorrent privacy [38]. The authors analysed the demography of BitTorrent content publishers and presented a highly skewed distribution of published content among them as well as the presence of a significant fraction of publishers located at hosting providers. This indeed validates some of our initial observations. In another study, Zhang et al. [118] presented the most extensive characterisation of the BitTorrent ecosystem. This study briefly examined the demography of the content publishers and showed a skewed distribution of the contributed content among them. The authors identify the publishers by their usernames. In Chapter 5 we show that this assumption may miss an important group of publishers who post fake content, i.e., fake publishers. In this thesis we go beyond the simple examination of demographics of content publishers. In Chapter 5 we identify, characterise and classify the major publishers and more interestingly reveal their incentives and their motivating business model.

2.2.7 Fake Content in BitTorrent

There are several studies presenting the possible threats in the Internet. In [120] authors state that 40% of all computers are infected by botnets and can be controlled by attackers. Another study [93] reports high presence of malware and spyware content in the Internet.

The research about security of BitTorrent protocol mainly focused on the weakness of the protocol itself rather than the content. Shneidman et al. [105] was the first

to demonstrate the vulnerabilities of BitTorrent protocol to free-riders which results in gaining unfair benefits by the malicious peer. In [86] the authors designed three different exploits which allow peers to achieve big download performance without benefiting to the system. In addition, in [89] they extended that work showing that free-riding can also be done without the presence of the seeder. Another exploit was presented and evaluated in [108]. In [54–56] the authors presented possible attacks which can be performed in BitTorrent. In [55, 56] authors describe leeching attacks which can be classified as connection attacks or piece attacks. In [54] the seed attacks (bandwidth and connection attacks) are presented. Moreover privacy issue of the BitTorrent were analysed in [38, 42].

Few previous works have studied the malware propagation through P2P systems [70, 83, 104, 119]. Specifically, Kalafut et al. [70] analyse LimeWire whereas Shin et al. [104] analysed Kazaa. These authors look at the problem from the content perspective instead of the fake publisher perspective used in this thesis. This avoids that they discover more sophisticated strategies as those reported in our study in which the content is not the malware itself but includes a link to the malware. Similar content-based approach is applied in FakeDetector program [15] that looks for fake hashes in DirectConnect hubs (central servers to which downloaders connect) and reports found fake content to users and hub administrators. Finally, the authors of [70] propose to filter those content with a specific size since most of the malware content has specifically this size. Unfortunately, this solution is not valid for BitTorrent. Instead, in this thesis we propose a more sophisticated solution in Chapter 6 (TorrentGuard) that provides early detection of fake content.

Chapter 3

BitTorrent Measurement Techniques

3.1 Introduction

BitTorrent is responsible for a major portion of the Internet traffic share [23] and is daily used by hundred of millions of users. This has attracted the interest of the research community that has thoroughly evaluated the performance and the demographic aspects of BitTorrent. Due to the complexity of the system, the most relevant studies have tried to understand different aspects by performing real measurements of BitTorrent swarms in the wild, this is inferring information from real swarms in real time.

Several techniques have been used in order to measure different aspects of BitTorrent so far. In this Chapter we present a survey of different measurement techniques that constitutes a first step in the designing the future measurement techniques and tools for analysing large scale systems.

The rest of the Chapter is structured as follows. In Section 3.2 we present a survey of the existing BitTorrent measurement techniques. Afterwards, we describe the main challenges that these techniques face and the solution to some of them in Section 3.3 before concluding in Section 3.4.

3.2 Measuring the BitTorrent Ecosystem

In this Section we describe the BitTorrent measurement techniques defined in the literature so far. We classify them into two main categories *macroscopic* and *microscopic* depending on the retrieved information. The former obtains demographic and high-level performance information whereas the latter gathers peer level perfor-

Property	Portal Crawling	Tracker Crawling	Peers Crawler	Own client/plugin
Category	Macroscopic	Macroscopic	Microscopic	Microscopic
Type of Information	Torrents	Demographics and High level performance	Peer Level Performance	Peer Level Performance
Cost of Crawler Preparation	Low	Medium	High	High
Scalability	Very High	High	Medium	Medium-Low
Obtained Details	Basic	Medium	Advanced	Very Advanced
Completeness of Torrent Population	-	High	Very High	Low

Table 3.1: Comparison of main BitTorrent measurement techniques

mance information. A summary of different techniques is presented in Tab. 3.1.

3.2.1 Macroscopic Techniques

The main objective of these techniques is to understand the demographics of the BitTorrent ecosystem: the type of published content, the popularity of the content, the distribution of BitTorrent users per country (or ISP), the relevance of the different portals and trackers, etc. Furthermore, the macroscopic measurements allow to study some performance aspects such as the ratio of seeders/leechers, the session time of the BitTorrent users, the arrival rate of peers, the seedless state (period the torrent is without seeder) duration, etc.

We classify the macroscopic techniques into two subcategories: *BitTorrent portals crawling* and *BitTorrent trackers crawling*.

1) BitTorrent portals crawling:

The (major) BitTorrent portals index millions of torrents in a structured way. Furthermore, they provide detailed information about each indexed torrent (typically) on a specific torrent web page. For instance, in the case of the Pirate Bay, the torrent web page associated to a torrent with an assigned torrent-id equal to i can be accessed through the url `http://thepiratebay.org/torrent/i` (see Fig.

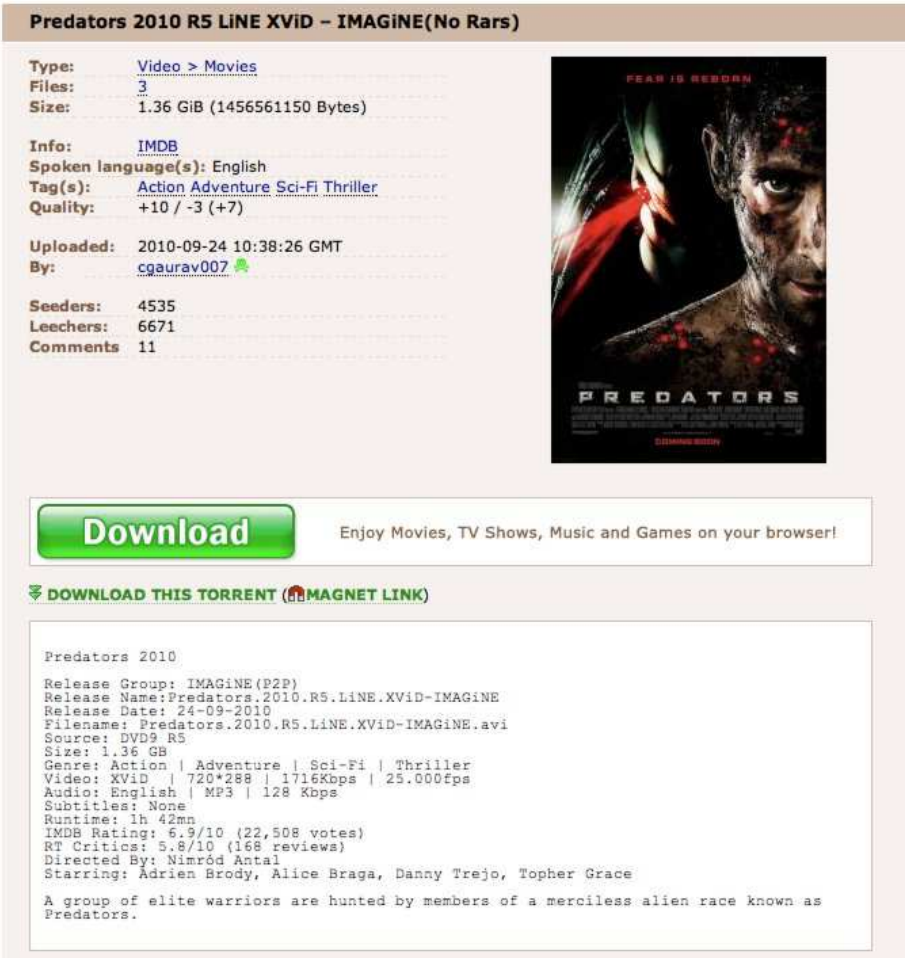


Figure 3.1: Example of the Pirate Bay torrent web page. HTML parsing techniques can retrieve the following information: Content name (Predators 2010 R5), Content category and subcategory (Video and Movies), Number of files (3), Size of the whole content (1.36 GB), Language (English), Upload date (2010-09-24), username uploading the .torrent file (cgaurav007), current number of seeders and leechers (4535 and 6671) and a text-box with further information regarding the content.

3.1). Hence, once we know the id assigned to a given torrent in the Pirate Bay, we just need to access its web page and parse it (using an html parser) to retrieve the torrent information. However, in order to analyse the demographics of BitTorrent we need to crawl a large number of torrents. Next we describe two types of crawling techniques that can be used in order to systematically crawl up to millions of torrents from a specific portal (we consider the Pirate Bay as an example):

- *Backwards Crawling*: In this case we aim to retrieve the information associated to the *alive* torrents published in the Pirate Bay from a given past date to the current instant. For this purpose our crawler sequentially parses all the torrents' web pages from the last published torrent (`http://thepiratebay.org/torrent/last_torrent_id/`) decreasing up to the first torrent published in the target date, for instance with torrent id k (`http://thepiratebay.org/torrent/k/`). The last published torrent-id can be identified either manually or using the RSS feed.

- *Upwards Crawling*: In this case we aim to retrieve the information associated to every torrent published in the Pirate Bay from now during a given time (e.g., one month). In this case, each new torrent will be assigned a torrent-id that can be learnt from RSS feed. We will use these learnt torrent-ids to crawl the torrents web pages.

By post-processing the retrieved data from the BitTorrent portal crawling we can characterise very relevant aspects of the BitTorrent ecosystem demographics. Next, we describe few representative examples. We refer the reader to [118] for a detailed analysis of the BitTorrent ecosystem demographics:

- *Content Popularity Distribution*: For this purpose we obtain the number of leechers and seeders for each specific torrent from the html parsing. Note, that if we want to study the evolution of popularity for a given torrent we have to periodically parse its web page to retrieve the evolution of the torrent population (i.e., number of leechers and seeders).

- *Distribution of number of published content per category and subcategory*: For this purpose we obtain the category and subcategory for each specific torrent from the html parsing.

- *Torrents Publishing Rate per date*: For this purpose we obtain the date when each specific torrent was uploaded from the html parsing.

By applying the described measurement study to different portals, we can perform a comparative study of the relevance of these portals in the BitTorrent ecosystem.

Finally, by tracking the evolution of the number of seeders and leechers for a given torrent we can also infer some performance metrics such as the seeder-to-leecher ratio and its evolution along the time.

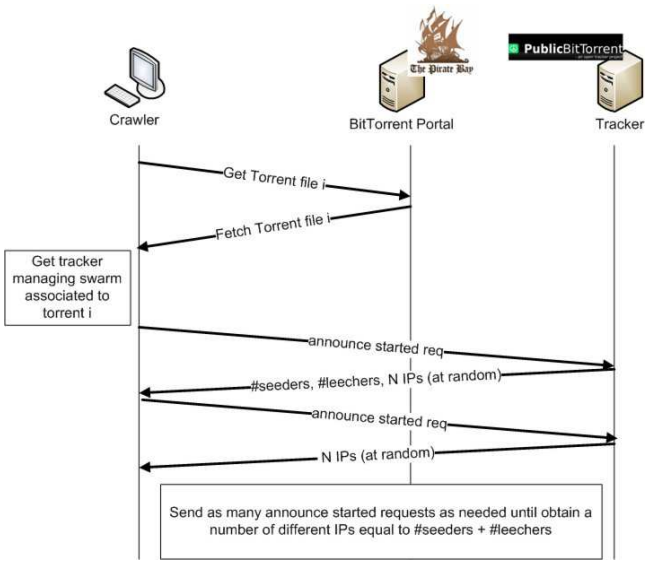


Figure 3.2: BitTorrent Tracker Crawler basic functionality: The BitTorrent crawler retrieves the .torrent file from a BitTorrent portal and obtains the IP address of the tracker managing the swarm from it. Afterwards, it sends as many *announce started* request as needed until obtain the IP addresses of all the peers participating in the swarm.

2) BitTorrent trackers crawling:

The crawling of a BitTorrent portal gives detailed information regarding the torrents (type, publishers) and some aggregated numbers such as the number of seeders and leechers. However, this does not suffice if we aim to study more detailed demographics parameters such as the distribution of BitTorrent users per country (or ISP) or relevant performance aspects such as peers arrival rate and peers session time. In order to study these issues we need to collect the IP addresses of the peers participating in the swarms. This can be obtained from trackers (remind that a tracker managing a given swarm knows the IP addresses of all the participants).

There are various ways of accessing the information of a tracker (i.e., IP addresses of participants in the swarms managed by the tracker):

- Getting access to the tracker logs [67]. This requires the tracker owner's collaboration.
- Using a tracker where the information is publicly available [62]. Unfortunately, only minor trackers offer this functionality.
- Using measurement techniques, i.e., crawling the tracker as depicted in Fig.

3.2. In this case we need to use a BitTorrent crawler that implements the part of the BitTorrent protocol to communicate with the tracker. More specifically, this crawler works as follows: first, we define the list of torrents whose participants' IP addresses we want to obtain. This list of torrents can be retrieved (for instance) from a BitTorrent portal. For each torrent in the list our crawler performs an initial *announce started* request to the correspondent tracker. From this request the crawler retrieves the number of participants (seeders and leechers) in the swarm and an initial list of IP addresses. Afterwards, the crawler performs as many *announce started* requests as needed to obtain as many IP addresses as the number of participants in the swarm.

Hence, by using any of the previous techniques we are able to collect the IP addresses of the participants in a large number of torrents. This data allows to study some relevant demographics and performance BitTorrent features. Next, we briefly describe some of them:

- *The Distribution of Clients per Country or ISP*: Some studies have applied the described crawling technique to a large number (even millions) of torrents [118]. Afterwards, the IP address of each client is mapped to its country and ISP (e.g., using the MaxMind database [18]). From this data we can compute the distribution of BitTorrent users per country and/or ISP.

- *Heavy Hitters*: By doing a cross-torrent inspection we can find those users (IP addresses) being present in a large number of torrents [38]. We name these users as *heavy hitters*.

- *BitTorrent Traffic*: The authors of [50] performed the described crawling technique in a short period of time (90 min.) over the most recent 40K torrents. This can be viewed as a snapshot of a portion of the BitTorrent ecosystem. By computing the traffic flowing between the BitTorrent clients in the different torrents, the authors estimate the Intra-ISP and Inter-ISP traffic generated by BitTorrent in a large number of ISPs.

- *Peers' Arrival Rate and Session Time*: If we apply any of the described techniques periodically on a given torrent, we are able to continuously monitor the peers participating in the torrent. Therefore, for each single user (i.e., IP address) we can approximately determine the instant in which it joins and leaves the torrent, thus being able to define the session time for each user. Furthermore by looking at the time between the subsequent arrivals of peers we can

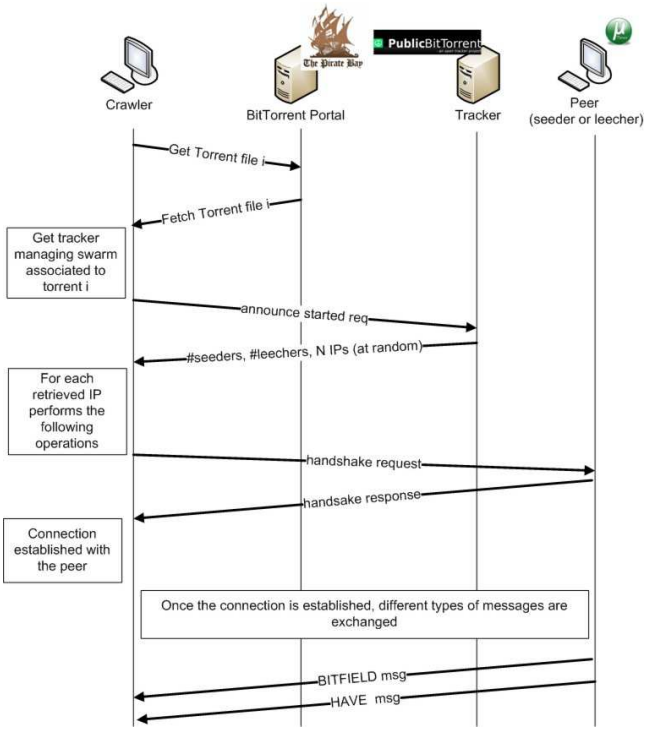


Figure 3.3: BitTorrent Peer Crawler basic functionality: The crawler retrieves the IP addresses of peers participating in a given swarm as explained in the macroscopic tracker crawling technique. Afterwards, the crawler contacts each individual peer, performs the handshake procedure and exchange different messages (BITFIELD, HAVE) to obtain different peer-level performance information.

infer the arrival rate. Authors of [62, 73] have performed this analysis in a large number of torrents.

3.2.2 Microscopic Techniques

The described macroscopic techniques retrieve exclusively the peers' IP addresses, thus only metrics associated to the presence/absence of the peer can be studied. Unfortunately, IP address does not suffice to infer relevant performance metrics at the peer level such as peers' download and upload rate. For this purpose we need to apply more sophisticated (but less scalable) techniques that we name *microscopic techniques*.

To perform microscopic techniques we need to implement different parts of the BitTorrent peer wire protocol. Any microscopic crawler has to implement the functions to perform the handshaking procedure. This is essential to connect to other

peers. The handshaking procedure can be done actively (the crawler initiates it) or passively (the crawler waits until a peer starts the handshaking). Once the crawler is connected to a peer, it exploits different messages of the peer wire protocol in order to measure different parameters. This process is illustrated in Fig. 3.3. Next, we describe the specific techniques proposed in the literature to measure the most important peer level performance aspects of BitTorrent:

Peer Type:

A BitTorrent peer can be of two types: seeder (has a complete copy of the file) or leecher (does not have a complete copy of the file). After the handshaking procedure succeeds with a peer, this one immediately sends a BITFIELD message to the crawler. By analysing the bitfield, the crawler classifies the peer as seeder or leecher [73, 107].

Furthermore, when using an active crawler there are some peers that do not respond to the crawler's handshake messages. These peers are typically located behind a NAT or a firewall that prevents the establishment of incoming connections. Thus, these peers are classified as *NATed* [73, 107]. In order to infer if a *NATed* peer is a seeder or a leecher we need to apply passive techniques and wait until the peer contacts our crawler.

Instantaneous Download Rate:

After the handshaking procedure is completed (either passively or actively) our crawler waits until it receives two HAVE messages from a given peer. The size of the chunk (e.g., 4 MB) used in a given torrent is well-known[1]. Furthermore, the crawler measures the time between the receptions of these two consecutive HAVE messages from a peer, that is approximately the time needed to download a chunk. Hence, by dividing the size of the chunk by the time needed to download it we can infer the instantaneous download rate of the peer. By repeating this operation periodically we can obtain the evolution of the instantaneous download rate of a given peer [107].

Average Download Rate:

In this case our crawler connects to a peer, obtains its bitfield and disconnects. After some time (e.g., 1 hour) the crawler repeats the same operation on the same peer. Then, by comparing the two bitfields, we can compute the number of downloaded chunks between the two connections to the peer. Since we know the size of each chunk (S), the number of downloaded chunks (D) and the time between the

[1]This information is available in the .torrent file.

two connections to the peer (T), we can easily compute the peer's average download rate as:

$$(S * D)/T. \tag{3.1}$$

Upload Rate:

This is probably the hardest parameter to be measured. Indeed, to the best of the authors knowledge there is no work that has properly measured the upload bandwidth. Rather, some few works have measured some parameters related to the upload rate. On the one hand, Isdal et al. [66] measure the physical upload capacity ($\geq$ upload rate dedicated to BitTorrent). For this purpose, the authors implement a passive crawler that measures the peers' upload capacity using the chunks sent by these peers to the crawler during optimistic unchokes. On the other hand, Siganos et al. [107] measure the number of IP packets sent by a node. For this purpose, the authors implement an active technique that uses a special type of ICMP message. The peers' answer to this ICMP packet includes the number of IP packets sent since the last time the computer was switched on. Hence, this crawler sends two of these ICMP packets separated a given time T. The answers to the first and second ICMP messages indicate a number of packets equal to $P1$ and $P2$. Therefore, the rate of IP packets sent by the peer is computed as:

$$(P2 - P1)/T \tag{3.2}$$

Note that this rate includes IP packets associated not only to BitTorrent but also to other applications.

Chunk distribution (Rarest First performance):

An important aspect of BitTorrent delivery mechanism is the Rarest First Algorithm. In order to study its performance we have to analyse how the distribution of the number of available copies of each chunk in a swarm looks like. For this purpose it is possible to implement a crawler that collects the bitfield of a large number of peers in a swarm (ideally all) in a relative short period of time (few minutes). By analysing the collected bitfields the objective is achieved, i.e., computing the number of available copies of each chunk in the swarm and calculating its distribution. This study was performed in [73] demonstrating that the Rarest First Algorithm guarantees a uniform distribution of pieces.

3.2.3 Complementary Techniques

Some researchers have used measurement techniques that can complement the macroscopic and microscopic techniques described above. On the one hand, some crawlers [73, 107] have implemented the DHT and/or PEX functionalities in order to learn the IP addresses of the peers participating in a given swarm. This can complement or even substitute the crawling of trackers explained as one of the macroscopic measurement techniques. On the other hand, some research groups have implemented their own BitTorrent client [27] or a plugins for a popular BitTorrent client such as Vuze [41]. These clients (or plugins) report information to a log server. This technique complements the microscopic measurements techniques since it gives very accurate information regarding peer level performance parameters, for instance it can precisely informs about the peer download and upload rate. On the downside, the scalability of this technique is limited to the number of clients running our BitTorrent client (or plugin). Moreover, the retrieved data is only representative for a specific client with a specific implementation, thus the obtained results may not be generalised.

3.3 Challenges

In this Section we enumerate the main challenges faced by the previously described techniques as well as possible solutions for some of them:

Peer Identification:

In BitTorrent the peers do not have a permanent Peer-ID. Every time a BitTorrent client is started a new random Peer-ID is generated. Then, it is not possible to follow a peer across multiple sessions using its Peer-ID. Most of the studies performed so far utilise the IP address or the IP address+port to identify a single user across multiple sessions. This works for all those users having a static IP address. However, most of the BitTorrent users are residential users with a dynamic IP address that is frequently changed by their ISP. Hence, identifying these peers by their IP addresses introduces inaccuracies in the obtained data.

One way of guaranteeing the correct identification of a peer across sessions is using measurement techniques based on the implementation of your own BitTorrent client/plugin. Each installed instance of the client has assigned a unique and permanent ID (different from the Peer-ID used in the swarms), which is used by the client to report the logs to the log server. Other option is to get access to the log of

private trackers. In most of the private trackers the users are required to register with a username and password. Each time a user initiates a session in the tracker it has to login, thus it can be uniquely identified across the sessions. Unfortunately, both described techniques have scalability limitations.

Crawler's IP address banned by the Tracker:

The described macroscopic tracker crawling technique may result in crawler's IP address being banned by the tracker. In some studies [38, 48, 118] the crawler performs a large-scale crawling by continuously sending *announce started* requests to a specific tracker for a large number (e.g., thousands) of torrents. Then, the rate of *announce started* requests is very high what is detected by the tracker. The reaction of the tracker is blocking the IP address showing this anomalous behaviour. Therefore the crawler has to limit the *announce started* requests rate to avoid being banned by the Tracker.

LeBlond et al. [38] describe a technique to avoid being banned while keeping a very high rate of *announce started* requests. The technique consists on sending an *announce stopped* just after the *announce started* request. Then, the tracker removes the IP address of the crawler from its log just after answering the *announce started* request. By using this simple technique, LeBlond et al. report that they are able to crawl up to 750K torrents in around 30 min.

A second option is using an anonymisation service such as TOR [25]. By using this service, the messages sent by the crawler pass through an overlay of proxies before reaching the tracker. Then, the IP address seen by the tracker is that of the egress node from the proxies overlay, thus the tracker cannot block the actual crawler's IP address.

Finally, we can increase the rate of requests to the tracker using several instances of the crawler distributed among different machines with different IP addresses.

Crawler's IP address blacklisted by the client:

In the case of microscopic measurements the crawler always performs the handshaking procedure with the target peer. Afterwards, it retrieves the needed information (e.g., the bitfield) and then it can either keep connected or disconnect and reconnect after a while. In the first case, since our crawler does not provide any chunk to the peer, due to the BitTorrent functionality, the peer is likely to substitute the crawler by other peer in its neighbourhood. Once the crawler has been removed from the peer's neighbourhood, it is typically hard to reconnect since the peer recognise the crawler as a useless peer who does not provide any data. In the second case

after the crawler connects and disconnects from a given peer few times (2 or 3), this peer also blacklists the crawler's IP address. The IP addresses in the blacklist have an associated timer and after this timer expires the IP address is removed from the blacklist. This means that the crawler can contact a given peer in the intervals $\geq$ blacklist timer[2]. However, sometimes we want to monitor the peers with a higher resolution than that imposed by the timer. In this case, we can use several instances of our crawler, each one with a different IP address and contact a given peer following a round robin schedule [73]. We could also use TOR, if two connections to the same destination are at least 10 min. apart, TOR establishes a new overlay path with a new egress node. Thus, TOR guarantees a 10 min. resolution.

Completeness of a torrent population

If we want to retrieve the complete population downloading a given torrent, we need to crawl all the trackers included in the .torrent file. Furthermore we have to retrieve the list of the peers that use the DHT instead of using a tracker. This crawling can be quite costly since some torrents can use tens of trackers.

Upload Rate Estimation:

We have discussed above the difficulties for measuring the upload rate and what other parameters have been measured as an approximation of the upload rate so far. In order to properly measure the upload rate of a peer we need to use a measurement technique based in our own BitTorrent client implementation.

3.4 Conclusion

In this Chapter we have presented and classified the main measurement techniques applied in order to understand different aspects of one of the largest-scale systems in the current Internet, i.e., BitTorrent. We believe that the described techniques can constitute the basis for the design of measurement tools for the analysis of current and future large-scale systems in the Internet, but also other environments. In the next Chapters we will use some of the above-mentioned techniques to make a detailed study about BitTorrent ecosystem with the emphasis on socio-economic factors.

[2]This timer value varies among the different clients. A conservative estimation based in our studies is 2 hours.

Chapter 4

Characteristics of BitTorrent Swarms

4.1 Introduction

BitTorrent is one of the most used application in the current Internet and is responsible for an important portion of the upstream and downstream traffic as revealed by recent reports [23]. The significant footprint of BitTorrent in the Internet has motivated researchers and practitioners to dedicate an important amount of effort to understanding and improving BitTorrent. However, despite this effort, we still have little knowledge regarding the connectivity properties exhibited by *real* BitTorrent swarms at both swarm and peer level. Due to the difficulty in collecting the required information from real swarms, most of the existing works that analyse connectivity properties are based on simulations [111] or experiments in controlled environments [30, 51]. As a result, they are likely to miss some of the effects affecting BitTorrent swarms in the wild. To the best of the authors' knowledge, there are just a few previous studies that evaluates few properties of the overlay topology (i.e., swarm level connectivity) of real BitTorrent swarms [61].

The analysis of the connectivity properties at the swarm level (i.e., overlay topology) and at the peer level (i.e., peers' neighbourhood composition) in real BitTorrent swarms can reveal important information such as: (i) the efficiency of a swarm to disseminate content [111], (ii) the resilience of a swarm to different events such as being partitioned [30], (iii) the efficiency and overhead generated by swarming and neighbour selection algorithms and (iv) the locality-bias exhibited by current BitTorrent swarms. Furthermore, the characterisation of these properties is of interest to: (i) researchers and practitioners designing (improving) new (existing) BitTorrent clients or related algorithms; (ii) companies using BitTorrent for critical functions such as software release, backups distribution or content replication [2, 3, 10, 17] and

(iii) Internet Service Providers (ISPs) carrying BitTorrent traffic.

In this Chapter, we first present a methodology to collect the connectivity information at both the swarm and the peer level for the entire lifespan of a real torrent. Specifically, we discover new torrents just after their birth by using the RSS service of the most important BitTorrent portal, namely the Pirate Bay. Afterwards, we exploit the Peer Exchange (PEX) extension of the BitTorrent protocol to gather the set of neighbours for each peer. PEX is a gossiping technique which main goal is to allow peers to exchange their list of neighbours so that they can learn about other participants in the swarm without contacting the tracker. Note, that PEX has been implemented by most of the existing BitTorrent clients and in particular by the most popular ones such as uTorrent or Vuze [14]. The information collected from PEX (i.e., a peer's neighbourhood) is the connectivity information at the peer level. Furthermore, by aggregating the neighbourhood information collected from every peer in a swarm we are able to build the overlay topology of that swarm (i.e., swarm level connectivity). We retrieve the information from each active peer every 10 minutes and then study the dynamic evolution of both: the overlay topology of the swarm and the composition of each peer's neighbourhood.

We have applied the described methodology to collect the connectivity information of 250 real torrents, including more than 150K peers, since their birth during a period of 15 days. This dataset constitutes the basis for our analysis, which is divided into two parts.

In the first part we analyse the connectivity properties at the swarm level. First, we perform a traditional graph theory analysis to understand the basic characteristics of the overlay topology of real BitTorrent swarms, namely clustering coefficient, characteristic path length and node degree distribution. We also analyse how the swarm size affects them. Furthermore, we compare these observed properties with those associated with well-known graph models such as random graphs, small-world or scale-free networks. Second, one of the most important features of a graph is its resilience. Specifically, in our study we evaluate the resilience of real BitTorrent swarms to be partitioned. To this end we consider two types of events: churn and a possible attack represented by a random node removal and a selective node removal processes, respectively.

In the second part of the analysis, we focus on the connectivity properties at the peer level. First, we study the variability in the composition of peers' neighbourhoods along time, paying special attention to the presence of stable neighbours. Second, we perform a thorough study in order to understand whether the level of locality observed in the composition of the peers' neighbourhood is higher/lower than

the expected from the random neighbour selection process implemented by default in BitTorrent. Furthermore, we group those peers presenting a significant positive deviation in the exhibited locality by their neighbourhoods into ISPs and countries. This simple technique allows us to discover ISPs and countries that are likely to be enforcing locality (e.g., using throttling techniques [58]).

Our main contributions and findings in this Chapter can be summarised as follows:

- We present a novel measurement technique to monitor the connectivity information of a real BitTorrent swarm during its complete lifespan.

- Our results demonstrate that, contrary to what previous studies claim [51, 61], real BitTorrent swarms are not random graphs. Furthermore, they are not small-world and thus they are not scale-free networks.

- Real BitTorrent swarms are fully resilient to be partitioned under a random removal process (e.g., churn). However, they are significantly less resilient than random graphs to a highest-degree node removal process (e.g., an attack). This result is of interest to researchers and practitioners that work on the enhancement of the different aspects (including resilience) of BitTorrent. Furthermore, our observation is also useful for those companies using BitTorrent for critical functions such as software release or content replication.

- Both leechers and seeders change a significant portion of their neighbours continuously. This is a consequence of the combination of the different neighbour selection algorithms implemented by current BitTorrent clients. This finding suggests that current BitTorrent clients incur a relatively high communication overhead that might not be necessary. Again, this result is useful for researchers and practitioners working on the improvement of BitTorrent.

- Leechers keep stable connections with a handful of other peers with which they exchange most of the traffic. This number of users is typically larger than the commonly used number of unchoke slots (4). Therefore, leechers tend to multiplex their resources (i.e., unchoke slots) to optimise their download performance. Furthermore, seeders typically do not keep stable connections with any peer so that they homogeneously distribute pieces among the participants in the swarm. Again, this result is useful for the design and validation of improvements on existing BitTorrent algorithms or the design of new algorithms.

- 45% of the analysed peers present at least 30% more local neighbours than expected from a pure random neighbour selection process. Furthermore, we observe an even more pronounced locality deviation in the set of stable neighbours (remember that the stable neighbours are those with which a peer exchange most of its traffic). This suggests that ISP locality enforcement policies (e.g., throttling), the proliferation of successful locality-aware P2P clients or plugins such as ONO [41] and other networking effects such as congestion are leading BitTorrent peers to exhibit a higher locality-bias in the composition of their neighbourhoods. This result is of interest to ISPs and to those researchers and practitioners working in the definition of locality-aware BitTorrent clients.

- Indian, large American and European ISPs present the largest proportion of high locality peers. Therefore, it is likely that these ISPs are enforcing locality (e.g., through throttling) among their BitTorrent clients.

The rest of the Chapter is organised as follows. Section 4.2 describes our measurement infrastructure and methodology as well as the dataset used along the Chapter. Section 4.3 presents our findings regarding the overlay topology of real BitTorrent swarms, whereas Section 4.4 evaluates the resilience of BitTorrent swarms and compares it with that observed for random graphs. Afterwards, Section 4.5 analyses the stability in both the overlay topology and the peer's neighbourhood composition along time. Section 4.6 studies the locality-biased composition of peers' neighbourhoods. Finally, Section 4.7 concludes the Chapter.

4.2 Measurement Methodology

The aim of our measurement study is to retrieve the graph topology of real BitTorrent swarms. For this purpose, we collect the neighbours list (or neighbourhood)[1] of each peer in the swarm by using the Peer Exchange (PEX) extension of the BitTorrent protocol. In the rest of the Section we provide a detailed description of both the measurement infrastructure and the methodology. For a full description of the BitTorrent ecosystem we refer the reader to [75] and [118].

[1]Note that in the rest of the Chapter we will use routing table, neighbours list, neighbours set and neighbourhood undistinguishable.

4.2.1 Measurement Infrastructure

Our measurement infrastructure is formed by 12 virtual machines, with a single public IP address each, installed in 3 different physical machines. One of the VMs acts as *Master* whereas the other 11 are *Slaves*. One the one hand, the *Master* is responsible for learning new torrents from a BitTorrent portal and contacting the tracker that manages the swarm associated with each torrent. Furthermore, the *Master* coordinates to which IP addresses (i.e., peers) each *Slave* has to connect at any moment. On the other hand, each *Slave* has a list of IP addresses (i.e., peers) to monitor. The *Slave* tries to connect to each one of these peers and to retrieve the peer's neighbours list among other information.

4.2.2 Measurement Methodology

The methodology of our measurements is based on the one presented in [48] and has several similarities with the methodology used in [61]. In order to learn new torrents we decided to use the Pirate Bay portal. This is the most important BitTorrent portal according to Alexa ranking[2] and some research studies [118]. The Pirate Bay offers an RSS service where each new torrent is announced as soon as it is uploaded to the portal. Our *Master* is subscribed to this RSS service, so that, it can discover new torrents after their birth. This guarantees that we will be able to crawl the full lifespan of a given torrent. The RSS service provides the *Master* with the .torrent file that includes the IP address of the tracker managing the swarm associated with the torrent along with other information not relevant to this Chapter. The *Master*, then, periodically queries the tracker with the maximum frequency allowed by this one (around 10 to 15 minutes) to avoid being blacklisted. Each reply from the tracker includes: the number of seeders (i.e., peers with a complete copy of the file), the number of leechers (i.e., peers with an incomplete copy of the file) and a random set (typically 200) of IP addresses of peers participating in the swarm. Furthermore, the *Master* is responsible for coordinating the *Slaves'* activity. The *Master* learns the IP addresses of peers within a swarm from the tracker and also from the *Slaves* as we will see later. The *Master* has to schedule the connection of the different *Slaves* to a given peer. The *Slaves* contribute no chunks to other peers, thus, if a *Slave* connects a few consecutive times to a given peer, the latter blocks the former. In order to avoid this, the *Master* schedules the connection to each individual learnt peer in a round robin fashion so that a given *Slave* only connects to the same peer once every 11

[2]http://www.alexa.com/topsites

connections (around 2 hours). This prevents any peer from blacklisting our *Slaves*.

Each *Slave* receives a list of IP addresses (i.e., peers) to connect to. The *Slave* connects to each of them that is not behind a NAT, and gathers the neighbours list for those peers supporting the Peer Exchange extension (PEX). PEX is an extension to the BitTorrent protocol that allows peers to exchange their neighbours lists. This reduces the load at the tracker since peers are able to learn about other peers without asking the tracker. In particular, *Slave* retrieves the list of *connected* and *disconnected* neighbours from the PEX messages. Note that the list of connected neighbours includes those nodes with which the peer has currently an established connection[3], i.e., the peer's current neighbours. Hence, in our analysis we use exclusively the list of connected peers and refer to them as peer's neighbours list (or neighbourhood). It is worth noting that most of the BitTorrent clients and particularly the most popular ones such as uTorrent and Vuze support PEX [14], thus we are able to retrieve the neighbourhood for almost every reachable peer. Furthermore, each *Slave* informs the *Master* regarding the IP addresses obtained through PEX. If any of these IP addresses is new, the *Master* adds it to the list of IP addresses to be crawled.

As mentioned earlier, there are peers that are behind a NAT and are not reachable, therefore if we fail to connect to a given IP address 5 times we declare this peer as *unreachable*. Furthermore, due to the churn phenomenon some nodes join and leave the swarm dynamically, so a reachable node may become unreachable, thus after 5 times failing to connect to a previously reachable node we consider that it left the swarm.

4.2.3 Dataset Description

We have applied the described measurement methodology to 250 consecutively published torrents, learnt from the Pirate Bay's RSS service from December 20th 2010 until January 4th 2011. From this set of torrents we were able to learn the neighbours list for more than 150K peers. Specifically, we are able to derive the evolution of the neighbourhood composition for each peer since, as explained above, we periodically retrieve every peer's neighbours list. Furthermore we map the peers' IP addresses to their country and ISP using the MaxMind Database [18]. Finally, it is worth mentioning that roughly 70% of peers are unreachable (i.e., are located behind a NAT) and thus, we cannot directly collect their neighbourhoods snapshots.

[3]Disconnected neighbours are nodes that the peer knew in the past but with which it does not have currently an established connection.

However, these unreachable peers appear as neighbours in other peers' neighbours lists. Hence, by post-processing the collected data we are able to reproduce (at least partially) the list of neighbours of those unreachable peers.

4.3 Overlay Topology of BitTorrent Real Swarms

The structure of a BitTorrent swarm directly impacts its efficiency to distribute pieces [111] as well as its resilience against different events such as churn or possible attacks [30]. Random graphs, small-world and scale-free networks are reference graph models with well-known properties. For instance, random graphs are known to be robust to possible partitions and churn (this is why the original design of BitTorrent aims to create random graphs), scale free-networks are vulnerable to attacks against the highly connected nodes (or hubs) and small-world networks have been demonstrated to be very efficient to disseminate information [46, 79].

In this Section we analyse the basic topology characteristics of the connectivity graph of real BitTorrent swarms and compare them with the characteristics of the aforementioned well-known graph models in order to get a better sense of essential aspects (e.g., resilience or dissemination efficiency) associated with real BitTorrent swarms.

4.3.1 Methodology

We represent a swarm as a collection of vertices (V) and edges (E). Each peer within the swarm is represented as a vertex, thus peer i is represented by v_i. Therefore $V = [v_1, v_2, ..., v_n]$, where n is the torrent population. Furthermore, $e_{ij} = 1$ if peer i and j are connected and 0 otherwise. Hence, the connectivity graph (or matrix) is the representation of E in the form of a matrix. Note, that in BitTorrent the connections are bidirectional, thus the connectivity matrix is symmetric. For each torrent in our dataset we have collected the neighbourhood of each reachable peer every 10 minutes; therefore, we are able to present a swarm's connectivity matrix evolution over time in 10 minutes intervals. Thus, we analyse both static and dynamic properties of the connectivity matrix.

For each snapshot of the connectivity matrix we calculate the following standard parameters used in graph theory studies:

- **Clustering coefficient (*C*)** is the average of local clustering coefficients for all

the vertices:

$$C = \frac{1}{n}\sum_{i=1}^{n} C_i \tag{4.1}$$

The local clustering coefficient of given vertex shows how close the neighbours of the vertex are to form a complete graph [113]. It can be expressed as:

$$C_i = \frac{2|e_{jk}|}{n_i(n_i - 1)} \tag{4.2}$$

where e_{jk} represents the total number of edges between two neighbours of peer i and n_i is the total number of neighbours of peer i.

- **Characteristic path length (*L*)** is the average length of the shortest path between each pair of vertices in the graph and is defined as:

$$L = \frac{\sum_{i=1}^{n}\sum_{j=1}^{n} d(v_i, v_j)}{n(n-1)} \tag{4.3}$$

where n is the number of vertices in graph and $d(v_i, v_j)$ denotes the shortest distance between vertices v_i and v_j.

- **Diameter (*DI*)** is the greatest length from the set of all the shortest paths between each pair of vertices:

$$DI = max\{d(v_i, v_j)\} \tag{4.4}$$

where $i, j \in [1, n]$.

- **Node Degree (D_i)** shows the number of edges that a vertex, v_i, has to other vertices. It can be expressed as:

$$D_i = \sum_{j=1}^{n} e_{ij} \tag{4.5}$$

In our analysis this is calculated as the size of the peer's neighbourhood.

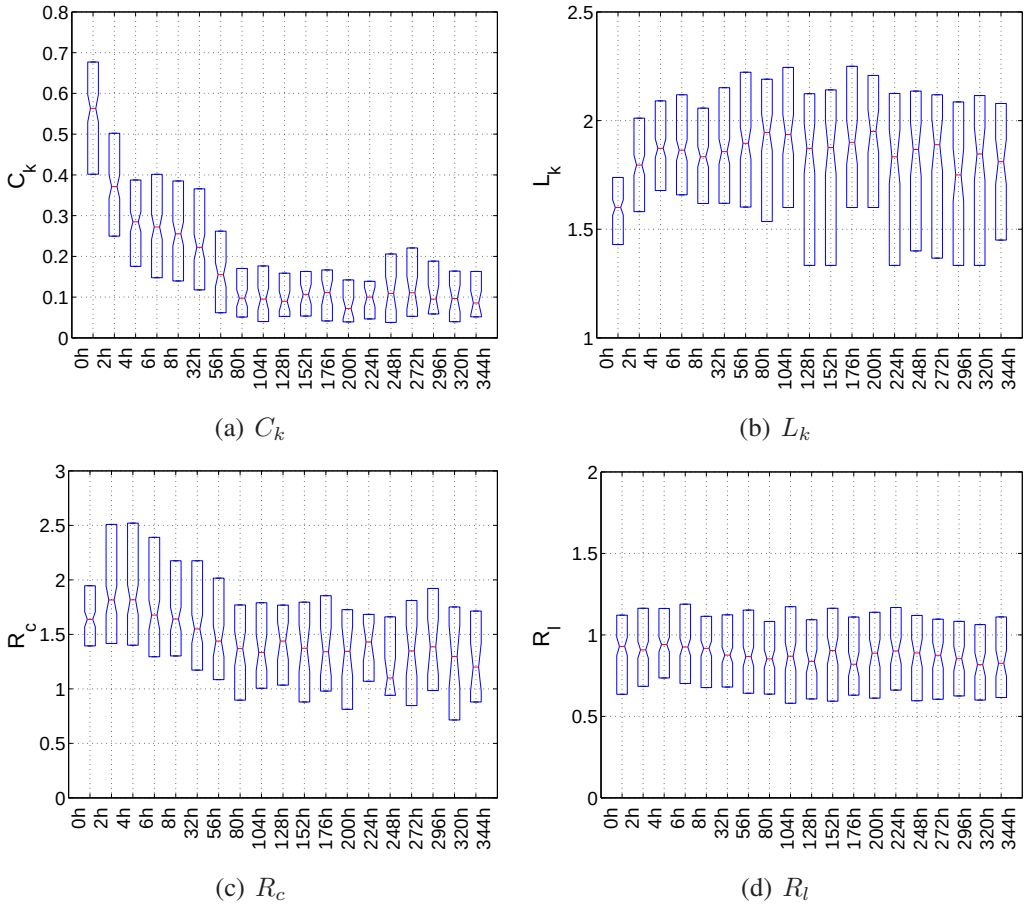
In the rest of this Chapter we use these parameters to analyse the static and dynamic characteristics of the overlay topology of real BitTorrent swarms. It is worth noting that since the results obtained from the analysis of *L* and *DI* lead to the same conclusions, in this Chapter we only discuss *L*.

4.3.2 Comparison to Small-World and Random Graphs

In this subsection, we study what type of graph model, small-world or random graph, fits better the overlay topology of real BitTorrent swarms. For this purpose, we apply a simple graph theory analysis. For each torrent in our dataset we calculate the clustering coefficient (C_k) and the characteristic path length (L_k) for all collected snapshots along the torrent lifespan. We also calculate the clustering coefficient (C_r) and the characteristic path length (L_r) for an equivalent random graph with the same number of edges and vertices as each analysed snapshot. These random graphs are created following an Erdos-Reni model [59] so that every possible edge in the graph exists with the same probability. Moreover, for each snapshot we also calculate $R_c = C_k/C_r$ and $R_l = L_k/L_r$. These two ratios indicate how similar a real swarm topology is to either a random graph or a small-world network. Those torrents having both R_c and R_l close to 1 would be random graphs whereas those torrents having R_l close to 1 and $R_c >> 1$ would present a small-world topology.

Figure 4.1(a) summarises the evolution of C_k for all the torrents in the dataset. In particular, it shows C_k for the first 8 hours of the torrent lifespan with a 2 hours difference interval and with a step of 24 hours after this point. For each of these instances of the torrent lifespan we present a box plot that indicates the 25, 50 and 75 percentiles of C_k considering all the torrents in the dataset. Figure 4.1(b) shows the same for L_k. These results help us to understand how the clustering coefficient and the characteristic path length evolve along time in real BitTorrent swarms. First, the clustering coefficient is higher in the birth phase of the torrent with values around 0.6 and continuously decreases to reach a stable state at the 56-80 hours after the torrent birth when the clustering coefficient is typically around 0.1 for most of the torrents. The explanation of this behaviour is the following: at the birth of the torrent the swarm size is small and most nodes are connected among them, thus leading to a high clustering coefficient. As time passes, the torrent population grows and then nodes are connected to just a portion of other nodes learnt from the tracker or through PEX. This produces a reduction on the clustering coefficient. The observed behaviour of the evolution of the clustering coefficient is consistent with previous emulation-based results [51], although the absolute values differ.

Second, the characteristic path length is smaller in the birth of the torrent with a median value of 1.6 and experiences a slight increment to reach a stable phase after few (4 to 6) hours where the median of the characteristic path length varies between 1.8 and 1.9. Again, in the birth phase we find a lower number of nodes that are well connected, which leads to a lower characteristic path length. However, even in



(a) C_k (b) L_k

(c) R_c (d) R_l

Figure 4.1: Distribution of C_k, L_k, R_c and R_l after x hours from the torrent birth for the 250 torrents from our dataset

the stable phase the characteristic path length is small, which guarantees that pieces of the file can easily reach any part of the swarm in less than 3 hops for the vast majority of torrents at any moment. This trend in the characteristic path length is also consistent with previous emulation studies [51].

In order to better understand whether the overlay topology of real BitTorrent swarms present similarities to random graphs or small-world topologies we present in Figure 4.1(c) and Figure 4.1(d) the evolution of the medians of the defined R_c and R_l ratios along time for all torrents in our dataset. We use the same box plot charts as for the clustering coefficient and the characteristic path length evolution. Note, that to obtain robust results, for each torrent snapshot in our dataset we generate 100 equivalent Erdos-Reni [59] random graphs of same size, and calculate the corre-

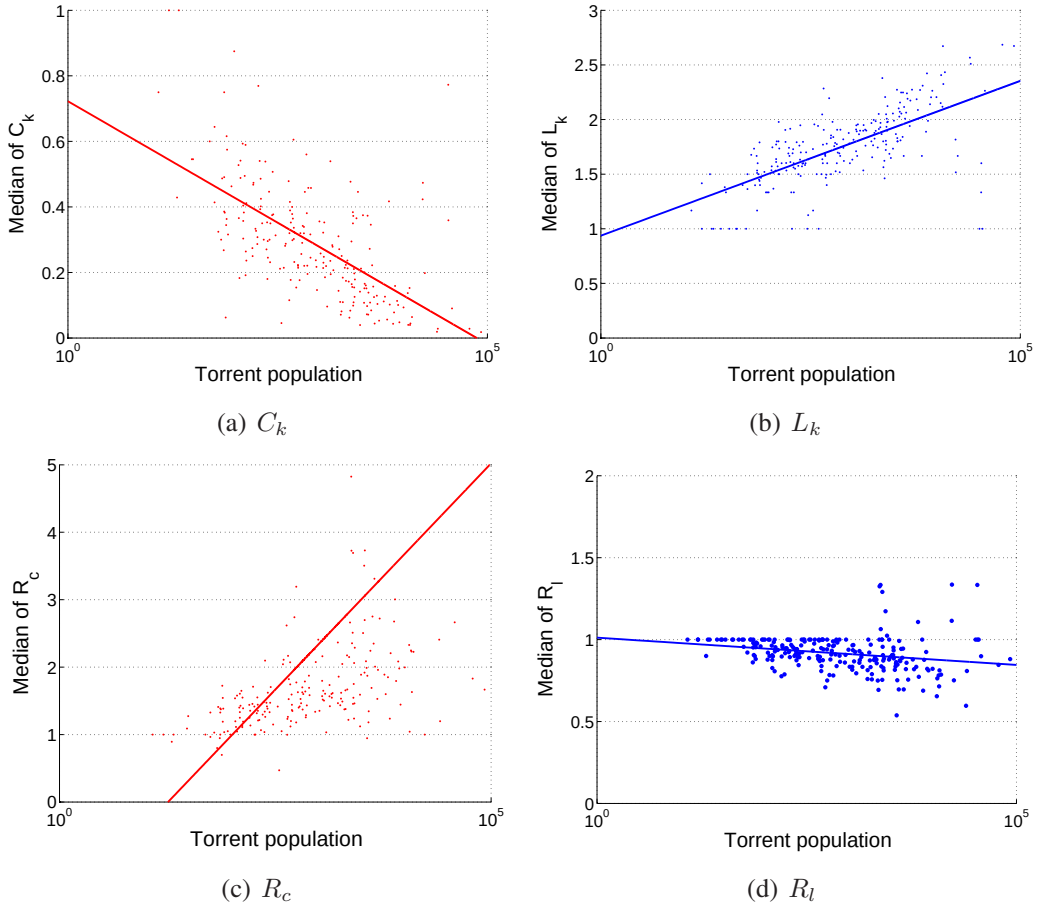
spondent 100 R_c and R_l values. In the figure we report for each snapshot the median R_c and R_l across the 100 samples. We observe that $1 < R_c < 2$ for most cases. This confirms that the clustering coefficient of the actual BitTorrent swarms is typically higher, although not significantly, than the one expected for an equivalent random graph of the same size. On the other hand, the median R_l is lower than 1 at every instance of the torrent lifespan for all the torrents in our dataset. Therefore, in real BitTorrent swarms we expect to have shorter paths between nodes than in equivalent random graphs of the same size. These results suggest that real BitTorrent swarms can be neither modelled as random graphs nor as small-world. These observations are contradictory to previous studies [51, 61], where the authors state that torrent swarms are random and close to random graphs, respectively.

4.3.3 Node Degree Distribution

In this subsection, we investigate the node degree distribution of real BitTorrent swarms. For this purpose, we study the distribution of our metric D for every snapshot in our dataset. Previous studies [51, 61] have analysed whether real BitTorrent swarms are scale-free networks. Note, that a graph is considered a scale-free network if the node degree distribution fits a power law distribution $P = ck^{-\gamma}$ where $2 < \gamma < 3$ [32]. Moreover, a scale-free network is by definition a small-world. Therefore, since in Section 4.3.2 we have proven that real BitTorrent swarms are not small-world we can safely conclude that they are neither scale-free networks. This result is aligned to those obtained by previous studies [51, 61].

However, it is still interesting to understand whether the node degree distribution of real BitTorrent swarms follows a power-law distribution, since this information reveal how unbalanced the connectivity among peers within a swarm is. To this end, first we have calculated the coefficients of a power-law function that better approximates the degree distribution of each analysed snapshot. Second, we perform a Kolmogorov-Smirnov (KS) test [90] to validate whether the power-law function obtained in the first step accurately approximates the real degree distribution of the snapshot[4]. Note, that we can only guarantee the power-law parameters estimation to be unbiased for those snapshots where the number of nodes $n \geq 50$. Therefore, to assure obtaining unbiased results, our analysis only considers those snapshots having at least 50 peers. The KS tests reveal that 80% of the considered snapshots present a degree distribution that can be accurately model by a power-law. All these snapshots

[4]The Kolmogorov-Smirnov test provides a positive result if the difference between the values of the two distributions at any given point is $< 5\%$.



(a) C_k (b) L_k

(c) R_c (d) R_l

Figure 4.2: Median value of the different graph metrics (C_k,L_k,R_c,R_l) per torrent vs. the total population of the torrent along its entire lifespan

present a value of $\gamma < 2$, confirming that none of the snapshots can be considered a scale-free network. Furthermore, the median and maximum values for γ across all these snapshots are 0.9 and 1.3, respectively. Therefore, there is a significant portion of real BitTorrent swarms that show a power-law distribution for node degree. We further explore this insight in the next subsection.

4.3.4 Impact of Torrent Popularity in the Overlay Topology

In this subsection we study whether the torrent popularity (i.e., the size of the swarm) affects the overlay topology of BitTorrent swarms. First, Figures 4.2(a) and 4.2(b) show the median of the clustering coefficient and the median of the characteristic path length as a function of the total number of peers that join the swarm

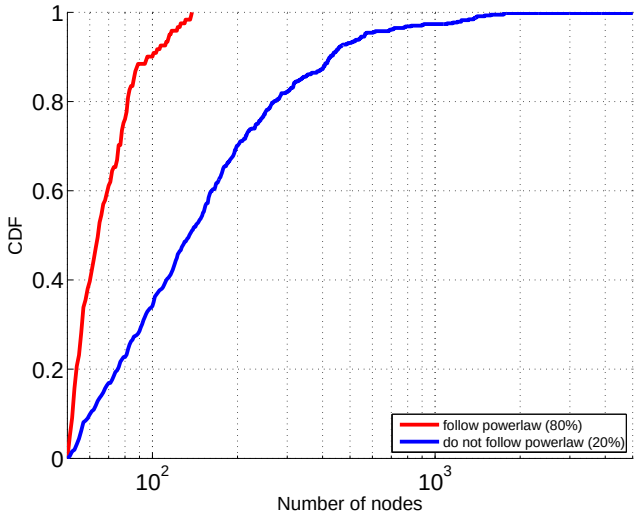


Figure 4.3: CDF of the size of snapshots presenting a power-law degree distribution (80%) vs. torrents not presenting it (20%). Note, that only snapshots with at least 50 peers are considered

along the whole measurement period for every torrent in our dataset, respectively. Moreover, each figure presents the polynomial curve fitting for the specific metric that visually shows the correlation between the metric and the torrent popularity. We observe a clear trend in which larger swarms typically present a lower clustering coefficient and a higher path length. This is an expected result since, as demonstrated by Figures 4.1(a) and 4.1(b), the clustering coefficient drops and the characteristic path length raises as the torrent population grows. This result is consistent with previous studies [61] in which the authors report that the clustering coefficient of popular torrents is small. However, the study of the absolute values of these metrics may lead to potentially wrong conclusions, e.g., the larger the swarm is the more random it is. Therefore, we have also studied the correlation between R_c and R_l and the size of the torrent. These results are shown in Figures 4.2(c) and 4.2(d) along with the corresponding polynomial fitting curves. Surprisingly, we observe a significant increase of R_c and a slight decrease of R_l with the size of the swarm. Therefore, relative and absolute metrics present opposite trends. Specifically, the reported trends for R_c and R_l show that the larger a swarm is the less random it (typically) is. It is worth mentioning that the analysis of these relative metrics allows us to correct previously reported results [51, 61] based on absolute metrics that concluded that popular torrents present a close-to-random graph topologies.

Furthermore, we have seen that 80% of torrent snapshots in our dataset present

a node degree distribution that follows a power-law distribution whereas the other 20% do not. It is also interesting to understand if the size of the swarm has any influence in the observed degree distribution. Toward this end, Figure 4.3 shows the CDF of the size of those snapshots presenting a power-law degree distribution along with the CDF of the size of those snapshots not presenting it. Remember that to guarantee unbiased results we consider only snapshots with at least 50 peers. The obtained results show that snapshots that have a power-law degree distribution are primarily small graphs ($\leq$150 peers), whereas large snapshots typically do not show power-law distributions. This suggests that large swarms tend to equalise the node degree among the participant peers.

4.3.5 Summary and Discussion

In this Section we have analysed the basic graph characteristics of real BitTorrent swarms, namely clustering coefficient, characteristic path length and node degree distribution. Our comprehensive study presents new results but also helps to shed light into controversial issues such as: (i) the random structure of BitTorrent swarms. Authors of [51] and [61] conclude that BitTorrent swarms are (close to) random graphs whereas [30] states that they are not; (ii) impact of the swarm size to its topology. The authors of [30] conclude that the size of the swarm does not impact its topology in their controlled experiments. Our results demonstrate that, contrary to conclusions of previous studies [51,61], real BitTorrent swarms cannot be modelled as random graphs. Furthermore, they are not small-world and thus they are neither scale-free networks. In addition, our results show that the popularity of a torrent (i.e., the swarm size) has a clear impact on the topological structure of BitTorrent swarms.

The higher clustering coefficient than equivalent random graphs along with the small characteristic path length shown by real BitTorrent swarms suggest that they present an efficient topology for disseminating chunks. However, there is some room for improvement; for instance making the structure of a real BitTorrent swarm more similar to a small-world topology may increase the efficiency in the information distribution as suggested by [51]. Another important aspect directly impacted by the overlay topology is the resilience of real BitTorrent swarms. Random graphs are by definition resilient topologies against different events such as churn. Following this principle, the default definition of the BitTorrent random neighbours selection procedure aims to generate random graphs in order to guarantee the resilience of the overlay topology. However, we have seen that the actual topology of BitTorrent

swarms cannot be modelled as a random graph. In the next Section we devote our effort to characterise and quantify the resilience of real BitTorrent swarms.

4.4 Resilience of Real BitTorrent Swarms

In the previous Section we have revealed the basic properties of the overlay topologies of real BitTorrent swarms and compared them to well-known graph models to get an intuition of relevant performance aspects (e.g., resilience, information dissemination efficiency) associated with real BitTorrent swarms. In this Section we deepen our investigation in one of the most relevant performance aspects, namely the resilience of real BitTorrent swarms. Resilience can be measured in different manners, in this Section we specifically analyse the resilience of real BitTorrent swarms to be partitioned and compare it to the resilience shown by equivalent random graphs, that are known to be resilient topologies.

4.4.1 Methodology

We analyse the resilience of a real BitTorrent swarm snapshot and an equivalent (same number of vertices and edges) Erdos-Reni [59] random graph to two type of events: (i) a random removal process in which we sequentially remove nodes selected at random until the graph is partitioned into (at least) two separated components; and (ii) a selective removal process in which we sequentially remove the node having the highest degree until the graph is partitioned into (at least) two separated components. The first event can represent, for instance, a scenario in which several (random) peers fail or leave the system (i.e., churn) whereas the second event can represent an attack against the swarm.

In order to quantify the resilience of a given graph to be partitioned we use two different metrics:

- K, represents the total number of peers that need to be removed in order to partition the graph.
- k, represents the percentage of peers that need to be removed to partition the graph. Therefore $k = 100\frac{K}{N}$ where N is the number of peers forming the swarm snapshot.

Note that we will refer to K_{real} (k_{real}) and K_{rand} (k_{rand}) as the values of K (k) associated with a real swarm snapshot and its equivalent random graph, respectively.

Due to the computational cost of these experiments we perform our study considering a random set of 400 snapshots in a stable phase (i.e., after the initial flash-crowd phase) from the 250 torrents forming our dataset. In addition to minimise the impact of abnormal peers (e.g., new incomers or nodes behind a NAT) we consider only those peers having a degree $\geq d$. We have perform the experiments with d = 2, 5, 10 and 15. Although the obtained results present some quantitative differences for different values of d, they are qualitatively consistent. Therefore, to make it simple, in this Section we present results for $d = 5$.

4.4.2 Random Node Removal

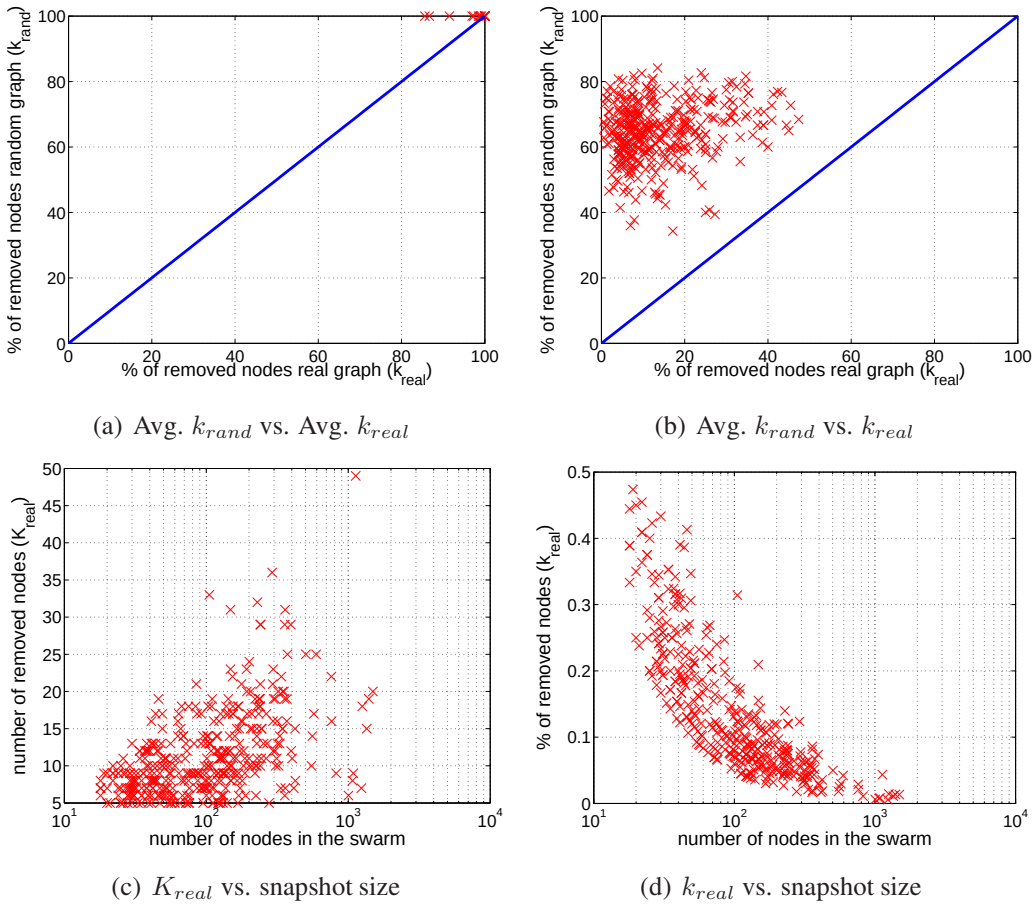
Given the random nature of this removal process we perform 10 simulation runs for each real swarm snapshot. Furthermore, for each snapshot we generate 10 Erdos-Reni random graphs with the same number of vertices and edges and for each of them we simulate 10 runs of the random removal process. Hence, for each snapshot and its equivalent random graphs we can calculate the average values of K and k as well as their standard deviations.

Figure 4.4(a) shows in a scatter plot the average value of k_{rand} vs. the average value of k_{real} for all the analysed swarm snapshots. We observe that the value of k for most of the analysed snapshots is located in the point (100,100). This means that both real snapshot and equivalent random graphs are fully resilient to a random removal process. There are only a few (2%) real snapshots that are not fully resilient, but even in these ones more than 85% of the nodes must be removed to partition the graph. Finally, it is worth to note that the standard deviation for both k_{real} and k_{rand} is negligible for all the studied snapshots.

Hence, the obtained results suggest that real BitTorrent swarms are in most of the cases fully resilient to a random removal process of peers that could be produced by different factors such as churn, peers' failures or an attack. This result is consistent with previous results obtained in a controlled environment [30].

4.4.3 Highest Degree Node Removal

In this case, the removal process for a given real snapshot is deterministic, therefore we perform a single run for each real snapshot. Again, we generate 10 different equivalent random graphs for each analysed snapshot. For each one of these random graphs we perform a single run of the removal process since as said before it is a deterministic process. Therefore, in this case we calculate the standard deviation



(a) Avg. k_{rand} vs. Avg. k_{real}

(b) Avg. k_{rand} vs. k_{real}

(c) K_{real} vs. snapshot size

(d) k_{real} vs. snapshot size

Figure 4.4: The metrics relation for the analysed swarm snapshots and their equivalent random graphs under a highest degree node removal process: Avg. k_{rand} vs. Avg. k_{real}, Avg. k_{rand} vs. k_{real}, K_{real} vs. snapshot size and k_{real} vs. snapshot size

only for K_{rand} and k_{rand}.

Figure 4.4(b) shows in a scatter plot the average value of k_{rand} vs. the value of k_{real} for all the analysed swarm snapshots. We observe that random graphs are significantly more resilient to the considered removal process than real snapshots. In particular, k_{rand} ranges roughly between 40% and 80% whereas k_{real} ranges between 1% and 50%.

Moreover, we analyse whether the size of the torrent has any impact in its resilience. Towards this end, Figure 4.4(c) and Figure 4.4(d) present scatter plots of the value of K_{real} and k_{real} vs. the snapshot size for every analysed real swarm snapshot, respectively. On the one hand, there is a surprisingly small correlation between

K_{real} and the size of the swarm that suggests that for large swarms the number of nodes to be removed to partition the graph is in the same order of magnitude than in the case of small swarms. This low correlation lead to the results observed in Figure 4.4(d) in which we see that the relative number of peers to be removed decreases significantly with the size of the swarm. Therefore, if we consider that the selective node removal process introduced in this Section represents an attack to a BitTorrent swarm, we conclude that the attacker would need roughly the same resources to perform an attack independently of the size of the swarm under attack. This observation is consistent with the fact that larger BitTorrent swarms are typically less random than small ones. Since random graphs are by definition resilient, the less similar the swarm structure is to a random graph the less resilient we expect it to be.

4.4.4 Summary and Discussion

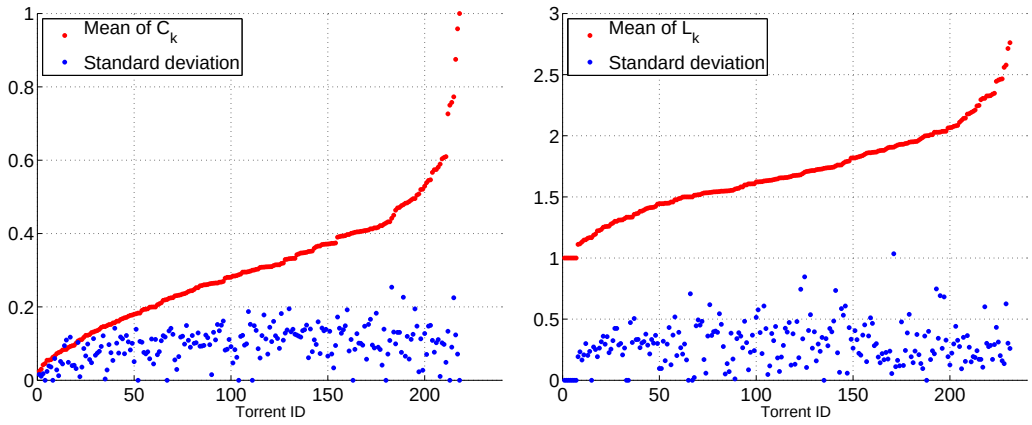
The results for the random node removal process are similar to the one obtained by means of emulation in [30], however, they significantly differ for the high-degree node removal process. The emulation results from [30] conclude that more than 80% of nodes must be removed in order to partition a BitTorrent swarm whereas our experiments reveal that the number of peers to be removed is $< 50\%$ for any of the real BitTorrent swarms snapshots analysed. Therefore, results in controlled environment seem to overestimate the resilience of BitTorrent swarms.

Our results suggest that BitTorrent swarms are fully resilient to a removal process of random nodes, however, their resilience to a selective removal process is significantly smaller than the one shown by random graphs. This result is of high interests to those companies that leverage the BitTorrent protocol to perform critical tasks such as software release or content replication since they can evaluate whether the resilience of BitTorrent to different events fulfil their requirements. Furthermore, researchers and practitioners working on the improvement of BitTorrent clients and protocol may find this result useful for the validation of their proposed solutions.

4.5 Stability of BitTorrent Swarms

In this Section we analyse the stability of BitTorrent swarms at following two different levels:

- *overlay topology stability* - we quantify the variability of the graph characteristics (clustering coefficient and characteristic path length) of a given torrent



(a) C_k (b) L_k

Figure 4.5: Mean an standard deviation of the clustering coefficient (C_k) and characteristic path length (L_k) for each torrent

over time.

- *peer's neighbourhood stability* - we quantify how stable the neighbourhood of a given peer is over time and carefully analyse the subset of stable neighbours.

4.5.1 Overlay Topology Stability of Real Swarms

We study the stability of the main topological parameters (i.e., clustering coefficient and characteristic path length) along the time. For this purpose, Figure 4.5 presents the mean and the standard deviation of the clustering coefficient (characteristic path length) for each torrent within our dataset sorted by the mean of C_k (L_k) in ascending order. We observe that for the major portion of the torrents the standard deviation is relatively small compared to the mean value for both the clustering coefficient and the characteristic path length. This suggests that overlay topologies of real BitTorrent swarms present a high stability.

4.5.2 Peer's Neighbourhood Stability

As explained in Section 4.2 we have collected the neighbourhood (i.e., list of neighbours) for each peer every 10 minutes approximately. Then, to quantify the peer's neighbourhood stability we have computed the percentage of neighbours that

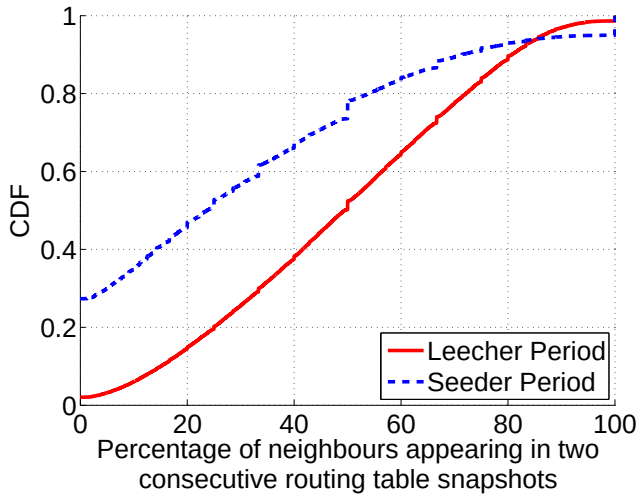


Figure 4.6: CDF of the percentage of neighbours that appear in two consecutive snapshots (10 minutes apart) of a given peer

appear in two consecutive neighbourhood snapshots for every peer in our dataset. In addition, we considered separately those periods in which a peer is a leecher and a seeder. Figure 4.6 presents the CDF for the defined metric. The results show that around half of leechers change 50% of their neighbours between two consecutive snapshots (i.e., every 10 minutes). This percentage dramatically increases up to 80% for seeders. Hence, BitTorrent peers are continuously changing a significant portion of their neighbours. This high variability is due to two causes: (i) the churn effect (i.e., peers leaving and joining the swarm) and (ii) the combination of different neighbour selection algorithms implemented in BitTorrent clients such as the unchoke algorithm, the optimistic connect algorithm (used in the leecher phase) and the optimistic disconnect algorithm (used in the seeder phase). Next, we briefly describe these algorithms:

4.5.2.1 Leecher Phase Algorithms

The *unchoking* algorithm makes a leecher select N (typically 4) neighbours to upload chunks to every 10 seconds. These neighbours are then *unchoked* whereas the rest of the node's neighbours are *choked* and will not receive data from the peer. The BitTorrent peer unchokes the N neighbours from whom it received the most data in the last 20 seconds. Therefore, the unchoking algorithm tries to keep *good* neighbours for exchanging traffic with. Furthermore, every 30 seconds the BitTorrent peer performs an *optimistic unchoke*. That is, it chooses a random neighbour

and uploads data to it. The optimistic unchoke allows nodes to discover better peers to exchange traffic with. Moreover, the most important BitTorrent clients such as Vuze or uTorrent utilise the *optimistic connect* [54] during the leecher phase. This algorithm drops the connection to those neighbours that have uploaded few or no data to the leecher during some time. These neighbours are substituted by new ones. The combination of the three described algorithms leads a leecher to identify and drop those peers from which the leecher is not obtaining enough performance, thus contributing to the reported high dynamism in the composition of peers' neighbourhoods.

4.5.2.2 Seeder Phase Algorithms

BitTorrent seeders apply different unchoke strategies depending on the implementation. The most extended strategies are: (i) *proportional* in which the seeder unchokes every 10 seconds the N leechers that have downloaded more data from it in the last 20 seconds and (ii) *balanced* in which the seeder unchokes peers following a round robin policy. Furthermore, seeders use the *optimistic disconnect* algorithm [54]. Based on this algorithm a seeder closes the connection to those peers to which it has not sent data for a long period of time (around 5 minutes). The combination of these algorithms (especially the balanced unchoking and the optimistic disconnect) aim to make the seeder communicating with as many peers as possible, rather than looking for *good* neighbours as occurs in the leecher state. As a result seeders show an extremely low stability in the composition of their neighbourhoods.

4.5.3 Stable Neighbours

In the previous subsection we have shown that both seeders and leechers are constantly changing a significant number of their neighbours. This leads to the reported high dynamism in the composition of peers' neighbourhoods. In this subsection, we change our focus and aim to analyse peers' *stable* neighbours. We define a stable neighbour as a neighbour that appears in all the neighbourhood snapshots of a given peer. Note that similar to the previous subsection we consider separately the leecher and seeder phases for a peer. As we have seen, the main algorithms applied by a BitTorrent leecher are: unchoke, optimistic unchoke and optimistic connect. The objective of these algorithms is to find a set of *good* neighbours that provides the highest possible download rate to the leecher and keep the interaction with them. Our hypothesis is that these algorithms converge to a set of *stable* neighbours with

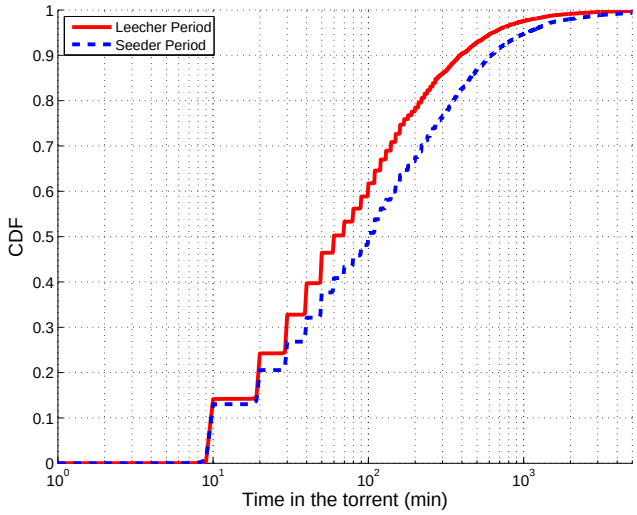


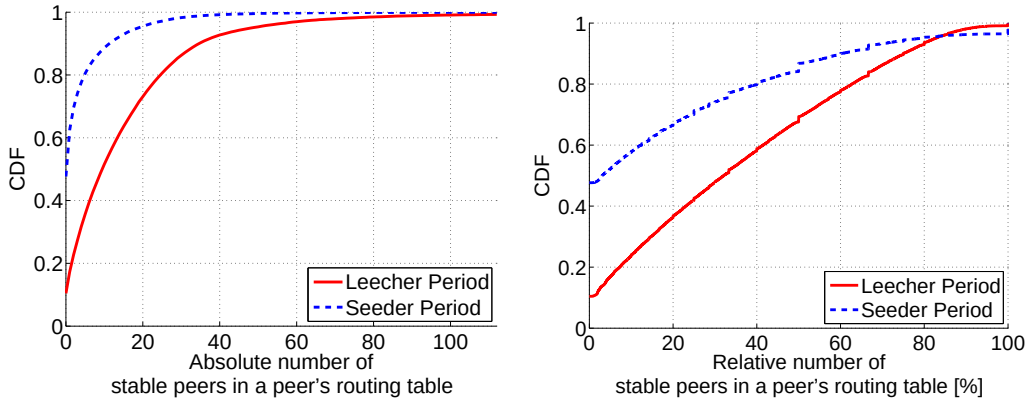
Figure 4.7: CDF of time in the torrent

which the peer systematically exchange traffic with. In addition, seeders apply different algorithms: *proportional* or *balanced* unchoke and optimistic disconnect. The final objective of the combination of these algorithms (especially, in the case of combining *balanced* unchoke and optimistic disconnect) is letting the seeder to distribute pieces of the content homogeneously among leechers. Therefore, our hypothesis is that during the seeder phase a peer tends to have a few (or none) stable neighbours.

In order to validate our hypothesis, we have collected all the neighbourhood snapshots for 50K peers within our dataset and analysed separately the leecher and seeder phases for each peer[5]. Figure 4.7 shows the CDF of the time that the analysed peers spend in their leecher and seeder phases, respectively. We observe that in median leechers stay in the system 70 min. whereas seeders stay 100 min. On the one hand, the leecher phase time roughly represents the download time, although in some cases the leecher leaves the torrent before finishing the download. On the other hand, we observe that seeders typically stay in the system longer than leechers. This result is supported by previous works that shown that users dedicated to do professional seeding (thus presenting long seeding sessions) are responsible for a major portion of the content published in the Pirate Bay [48].

Moreover, for each peer (and phase) we have calculated two metrics: (i) the number of stable neighbours and (ii) the percentage of stable neighbours as the ratio between the number of stable neighbours and the median size of the peer's neighbourhood in the considered phase. Figures 4.8(a) and 4.8(b) depict the CDF

[5]Note that some peers can present only one of the two phases.



(a) Absolute (b) Relative

Figure 4.8: CDF of the absolute number and percentage of stable neighbours in a peer's routing table

for these two metrics for the 50K analysed peers, respectively. The results validate our hypothesis.

Figure 4.8(b) shows that leechers keep an important percentage (30% in median) of *stable* neighbours. These are neighbours with which the peer systematically exchanges traffic. Furthermore, seeders have a much lower percentage of stable neighbours, in fact half of the seeders do not have any stable neighbour.

If we analyse now the number of stable peers (Figure 4.8(a)), we observe that leechers have in median 10 stable neighbours. This value is higher than the typical number of unchoke slots used by the leecher (4). This observation suggests that current BitTorrent implementations lead leechers to multiplex their resources (i.e., unchoke slots) in time so that they are able to attract a larger number of peers to obtain pieces from. Moreover, 60% of seeders presents a number of stable peers ≤ 1. This finding suggests that current implementations of seeding algorithms lead seeders to communicate with as many peers as possible. By doing so, seeders aim to obtain a homogeneous dissemination of pieces among participants in the swarm (avoiding selfish behaviours by which a peer tries to retrieve all the pieces from the seeder without contributing pieces to other peers).

4.5.4 Summary and Discussion

First, we have shown that both leechers and seeders continuously change a significant portion (more than half) of their neighbours. This results in a communi-

cation overhead that may be unnecessary, especially if BitTorrent is used in low resources devices. Surprisingly, our results reveal that the overlay topology properties (i.e., clustering coefficient and characteristic path length) of real BitTorrent swarms remain stable, despite the high dynamicity reported at the peer's neighbourhood level.

Second, our analysis of the set of stable neighbours leads to the following conclusions: (i) leechers tend to interact (i.e., exchange data) with a reduced number of neighbours, that is typically larger than the number of unchoke slots, in order to optimise their download rate, (ii) seeders present a very reduced number of stable neighbours, thus they interact with a large number of peers in order to guarantee the proper dissemination of pieces within the swarm.

The results reported in this Section reveal some interesting properties regarding the swarming efficiency driven by current BitTorrent client implementations and constitute a solid basis to design future BitTorrent implementations and compare their performance with the existing ones. Specifically, future developments can consider aspects such as: (i) reducing the dynamism in the formation of peer's neighbourhood thus reducing the associated communication overhead, and (ii) exploring the effects of reducing/increasing the number of stable neighbours for seeders and/or leechers.

4.6 Analysing Locality in Real BitTorrent Swarms

The random bootstrapping used in P2P applications, and more specifically in BitTorrent, is unnecessarily pushing a lot of traffic to the transit links of ISPs increasing their operational costs [36, 72]. Some ISPs have started to implement different policies (e.g., throttling) in order to minimise the impact of BitTorrent traffic in their networks [58]. Moreover, the research community has proposed promising solutions such as ONO [41] or P4P [115] to make a BitTorrent node select (when available) neighbours within its own ISP, thus confining as much BitTorrent traffic as possible within the ISP. In addition, other aspects such as network congestion can also affect the amount of BitTorrent traffic localised within an ISP. Note, that all these factors (e.g., ISP policies, research solutions or congestion) are likely to affect just a subset of peers in a swarm rather than the whole swarm. Therefore, in this Section we study the locality phenomenon at the peer level. Specifically, we investigate whether the locality level exhibited by a peer's neighbourhood and/or set of stable neighbours is higher/lower than the expected from the random neighbours

selection process implemented by default in BitTorrent.

4.6.1 Methodology

In this Section we consider the same 50K peers used for our analysis in Section 4.5.3. We first study whether the neighbourhoods of BitTorrent peers present an ISP- or a country-biased composition. That is, whether the number of neighbours from the same ISP (or country) as the peer is higher than the expected compared to a purely random process. Then, we repeat the analysis considering exclusively the peers' stable neighbours (i.e., neighbours with which the peer systematically exchange traffic with). Note, that we use the MaxMind database [18] to map each peer to its ISP and country. Next we describe our methodology, which is based on the methodology presented in [50]:

Let us consider a peer p belonging to a torrent swarm T. We denote *V(T)* as all peers participating in T. We also define *V(I,T)* as a subset of *V(T)* which includes all peers belonging to p's ISP (*I*) and *V(C,T)* as a subset of *V(T)* which contains all peers belonging to the same country (*C*) as p.

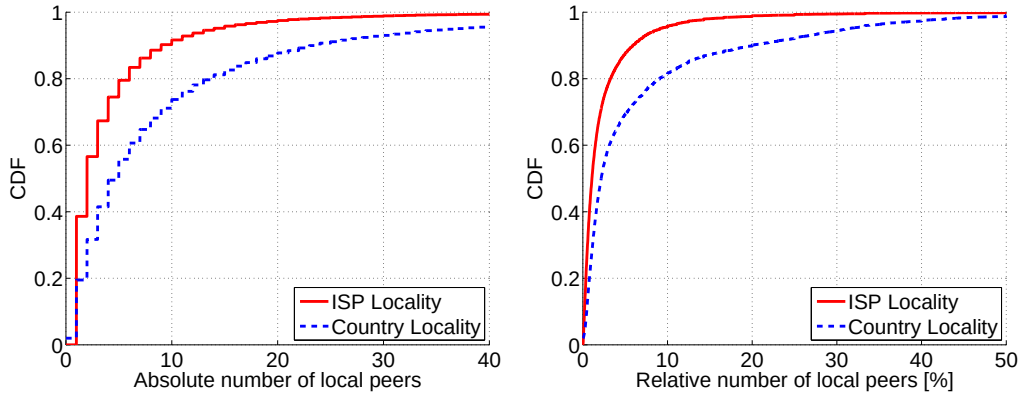
On the one hand, we consider a random neighbours selection hypothesis that represent the expected functionality of the BitTorrent protocol. We refer to this hypothesis as H_0 in the rest of the section. In particular, we calculate the expected (i.e., average) number of local neighbours that p should have from its ISP (E_i) and its country (E_c) in each of its neighbourhood snapshots under H_0. This is given by the mean of the Hyper-Geometric distribution[6].

On the other hand, we calculate the actual number of local nodes from the same ISP (I_n) and from the same country (C_n) that appears in p's neighbourhood.

Finally, we define a simple metric named the *Locality Ratio* (LR) that captures whether the neighbourhood of a given peer is biased towards having more local nodes than expected from a random selection process. More specifically, we define LR_I (ISP Locality Ratio) as I_n/E_i and, LR_C (Country Locality Ratio) as C_n/E_c. Hence, a peer with an $LR_I > 1$ and $LR_C > 1$ has a higher number of neighbours from its ISP and country than expected under H_0, respectively.

Prior to presenting our results, we apply a filtering technique to avoid a bias in the obtained results. Previous works [50, 64] reported that a given peer p can be

[6]The probability of getting x "successes" (i.e., local nodes) when drawing randomly W samples from a pool of N items, out of which M are "successes" is given by the HyperGeo(x, N, M, W). In our case, for a given peer N is represented by the swarm size - 1 (itself), M is represented by the number of local nodes (from the ISP or Country) -1 (itself) and W is represented by the peer's neighbourhood size.



(a) Absolute (b) Relative

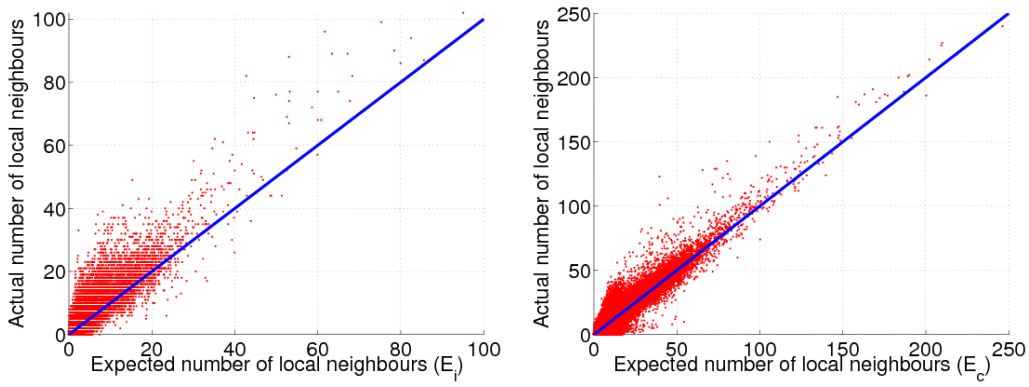
Figure 4.9: Number of local available nodes for peers in unlocalised torrents: *(a)* absolute and *(b)* relative [%]

located in an *unlocalisable* torrent[7] (i.e., swarm snapshot). Locality is, by definition, (almost) impossible for this peer since the number of local nodes in the considered swarm snapshot is 0 or very low. Therefore, we have removed from our dataset all those peers located in unlocalisable swarm snapshots. Specifically, we consider that a swarm snapshot is unlocalisable for a peer p if: (i) there are no other local peers or (ii) p's E_i (E_c) < 1 (i.e., the expected number of local nodes in p's neighbourhood is very low). To validate our filtering technique we have measured the absolute and relative (as a percentage of whole population) number of local nodes for those filtered peers. The results are shown in Fig 4.9. For the case of ISP locality we observe that (in median) there are just 1.5 local nodes for the filtered peers. Furthermore, these local nodes represent less than 2% of their torrents population. The results for country locality are similar. Therefore, we can conclude that our filtering technique successfully removes peers located in unlocalisable torrents.

4.6.2 Locality-biased Peer's Neighbourhood

In this subsection we apply the previously described methodology to every neighbourhood snapshot of the 50K analysed peers. Figure 4.10 shows E_i vs. I_n and E_c vs. C_n for every considered neighbourhood snapshot. We observe that most peers have a locality-ISP biased neighbourhood ($I_n > E_i$) whereas this bias is

[7]Note that the notion of unlocalisability applies to a peer. Therefore we can have peers for which the torrent is unlocalisable and others for which it is not, within the same swarm.



(a) ISP (b) Country

Figure 4.10: Expected number of local neighbours vs. actual number of local neighbours: *(a)* ISP and *(b)* Country locality

slightly lower when we consider the country criteria. Therefore, we can conclude that a significant portion of BitTorrent peers presents a locality-bias neighbourhood composition. To gain more insight into this phenomenon we next quantify the reported bias. To this end, Figure 4.11 presents the distribution of the median LR_I and the median LR_C of each peer across all its neighbourhood snapshots. We can observe that an important portion of peers (45%) have a surprisingly high $LR_I > 1.3$. This means that they have 30% more local neighbours from its own ISP than expected under H_0. This percentage gets reduced when looking at the locality at the country level where only 27% of peers shows $LR_C > 1.3$.

Furthermore, it is interesting to analyse the demographics of the observed locality phenomenon in order to discover whether there are ISPs presenting a large number of peers with a high level of locality. Towards this end, for each ISP in our dataset we collect the absolute and relative number of peers having a high LR_I. We refer to these peers as *high locality* peers. Specifically, we consider a peer as a *high locality* peer if it has a $LR_I > 1.3$. Table 4.1 shows the 10 ISPs with the largest number of *high locality* peers. In addition, the table reports the percentage of *high locality* peers and the median LR_I of the *high locality* peers for each one of these 10 ISPs. Interestingly, we observe the presence of major US and European ISP, such as Comcast (US) or Virgin Media (UK), in the list. This suggests that some major ISPs are implementing policies to reduce the transit traffic generated by BitTorrent that have a clear impact in the neighbourhood composition of their peers. Furthermore, it is worth mentioning the presence of 4 different Indian ISPs in the list. Therefore, the usage of techniques to reduce the transit traffic generated by BitTorrent seems to

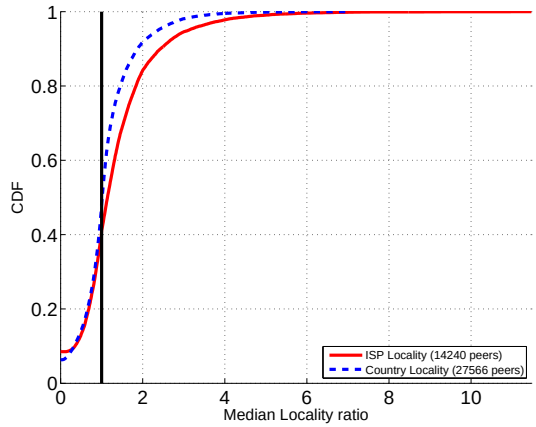
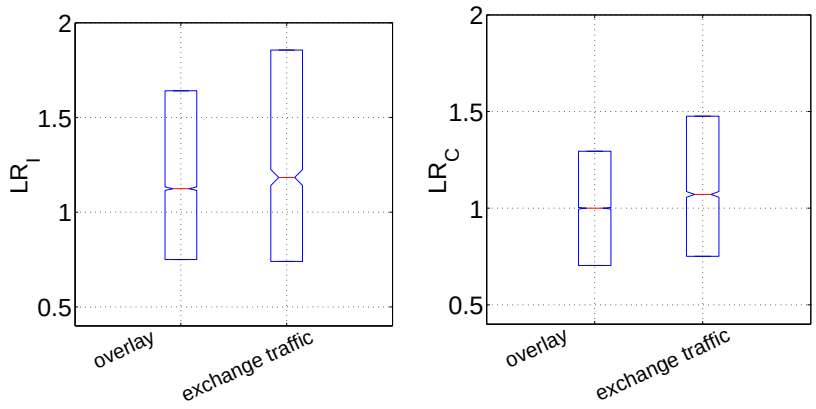


Figure 4.11: CDF of locality ratio

be common among major Indian ISPs as well.

We repeat the same analysis at the country level for peers with $LR_C > 1.3$. The results are shown in Table 4.2. India is the country with the highest number of *high locality* peers at the country level. The US occupies the second position in the ranking and we also observe the presence of some European countries. It is also worth noting that more than 60% of users from Taiwan are *high locality* peers. These results are consistent with the conclusion obtained at ISP level.



(a) ISP (b) Country

Figure 4.12: Distribution of LR_I and LR_C at overlay construction and the exchange traffic levels

ISP	Median	%
Bharti Broadband	2.22	79.75
NIB (National Internet Backbone)	1.77	42.40
Comcast Cable	1.65	36.80
PTCL Triple Play Project	1.89	55.44
CHTD, Chunghwa Telecom Co., Ltd.	1.89	72.19
Road Runner	1.63	36.27
Mahanagar Telephone Nigam Ltd.	1.97	42.86
RELIANCE COMMUNICATIONS	1.71	38.06
Virgin Media	1.72	38.16
SBC Internet Services	1.74	41.34

Table 4.1: ISPs with the highest number of *high-locality* peers at the overlay construction level

Country	Median	%
IN	1.89	25.38
US	1.58	29.51
GB	1.66	32.71
RU	1.80	23.15
PK	1.85	47.40
CA	1.60	31.95
TW	1.85	60.96
PL	1.71	36.94
FR	1.63	37.71
SE	1.58	15.83

Table 4.2: Countries with the highest number of *high-locality* peers at the overlay construction level

4.6.3 Locality-biased Peer's Stable Neighbours Set

The ultimate objective of the different policies implemented by ISPs [58] as well as researchers' proposals [41, 115] is reducing the amount of traffic crossing ISPs' transit links. Therefore, it is interesting to examine whether we observe any *locality* effect at the traffic-exchange level. For this purpose, we apply the described methodology to the stable neighbours of the 50K analysed peers. Remind that the stable neighbours are those nodes with which a peer systematically exchanges traffic with.

Figure 4.12 shows box plots with the distribution of LR_I and LR_C at both the neighbourhood and the stable neighbours (i.e., exchange traffic) levels. Specifically, the boxes represent the 25, 50 and 75 percentiles of the different LR distributions. First, we observe that the set of stable neighbours shows a slightly higher locality-bias than the neighbourhood at both ISP and country levels. Second, the figure confirms that the locality effect is more marked at the ISP level than at the country level.

Finally, we have repeated the demographic analysis considering in this case the set of stable neighbours. The results are presented in Tables 4.3 and 4.4 for ISP level and country level locality, respectively. We conclude that the observations done at the neighbourhood level are also valid at the stable neighbours level: the presence

ISP	Median	%
NIB (National Internet Backbone)	1.93	51.84
Bharti Broadband	2.53	76.52
CHTD, Chunghwa Telecom Co., Ltd.	1.69	61.11
Comcast Cable	1.63	31.34
Telecom Italia	1.65	30.91
Bredbandsbolaget AB	1.55	45.83
Telefonica de España	1.57	25.64
Road Runner	1.77	36.00
PTCL Triple Play Project	2.28	41.18
Mahanagar Telephone Nigam Ltd.	2.47	66.67

Table 4.3: ISPs with the highest number of *high-locality* peers at the traffic exchange level

Country	Median	%
IN	1.93	50.83
US	1.70	30.15
PL	1.83	59.32
RU	1.76	20.43
GB	1.83	36.44
SE	1.55	13.00
TW	1.71	48.00
DE	1.56	21.65
ES	1.58	9.94
CN	1.68	30.91

Table 4.4: Countries with the highest number of *high-locality* peers at the traffic exchange level

of several major ISPs from India and several major US (Comcast) and European (Telecom Italia, Telefonica España) providers among the top 10 ISPs with a larger number of *high locality* peers.

4.6.4 Summary and Discussion

In this Section, we have demonstrated that a significant fraction of the studied peers (45%) present at least 30% more local neighbours than those expected from a purely random neighbour selection process. Furthermore, this biased composition is even more marked when we consider the set of stable neighbours, which are those with which a peer exchange most of its traffic. The enforcement policies implemented by ISPs (e.g., throttling) along with the proliferation of successful locality-aware BitTorrent clients such as ONO [41] and other network effects such as congestion seem to be the cause for the observed results. Furthermore, our results reveal that Indian ISPs along with large American and European ISPs are those hosting a larger number of users presenting a higher locality-biased in their neighbourhood composition. Finally, our results show a higher locality-biased at the ISP than at the country level. This can be explained since it is likely that in a specific country just some ISPs implement locality-enforcing techniques, therefore when analysing

the locality-biased at the individual ISPs we observe a larger bias than in the case of considering aggregately all the peers within the country.

The obtained results are useful for ISPs as well as for the evaluation of novel locality-aware BitTorrent implementations under development by researchers and practitioners.

4.7 Conclusions

In this Chapter we present a comprehensive study of the overlay topology structure and the connectivity properties at peer level of real BitTorrent swarms. For this purpose we leverage information collected from 250 real torrents and more than 150K peers. Our results demonstrate that real BitTorrent swarms present a relatively efficient topology for the dissemination of information and are resilient to churn (i.e., random node removal process). However, real swarms are significantly less resilient to possible attacks (i.e., highest-degree node removal process) than equivalent random graphs. Furthermore, our analysis of the composition of peers' neighbourhoods reveal that current BitTorrent implementations make both leechers and seeders modify a significant portion of their neighbourhoods in short periods of time. This leads to a communications overhead that might not be needed. In addition, a leecher (typically) keeps stable connections with just a handful of its neighbours with which it exchange most of its traffic. In contrast, seeders do not keep long-term connections with other peers in order to guarantee the homogeneous distribution of pieces among the participants in the swarm. Finally, our results reveal that a significant fraction of peers present a clear locality-biased composition of both their neighbourhoods and their set of stable neighbours. This suggests that locality-enforcing policies of some ISPs, the proliferation of locality-aware BitTorrent clients and some other network effects such as congestion are localising an important part of BitTorrent traffic within some ISPs. In particular, our results show that the ISPs hosting a larger portion of peers with an important locality-biased neighbourhood composition are large US, European and Indian ISPs.

These insights seem to be of interest and usefulness to: researchers and practitioners working on the improvement of BitTorrent related algorithms, companies that use BitTorrent to perform critical tasks such as software release or content replication and ISPs carrying BitTorrent traffic.

Chapter 5

Publishing Content in BitTorrent

5.1 Introduction

Peer-to-Peer (P2P) file-sharing applications, and more specifically BitTorrent, are clear examples of *killer* Internet applications of the last decade. However, the socio-economic aspects of other P2P file sharing systems in general and BitTorrent in particular have received little attention. More specifically, a key factor in the popularity of BitTorrent is the availability of popular and often copyrighted content (e.g., recent TV shows and Hollywood movies) to millions of interested users at no cost. This raises an important question about the incentive of publishers who make these files available through BitTorrent portals. To our knowledge, prior studies on BitTorrent have not addressed this critical question.

In this Chapter, we study content publishing in BitTorrent from a socio-economic point of view by unravelling *who* publishes content in major BitTorrent portals, and *why*. Toward this end, we conduct a large-scale measurement over two major BitTorrent portals, namely Mininova [19] and the Pirate Bay [21], to capture more than 55K published content objects that involve more than 35M IP addresses. Using this dataset, we first examine the contribution of the individual content publishers and illustrate that a small fraction of publishers (3%) are responsible of uploading 67% of the published files that serve 75% of the unique peer downloads in our major dataset. Furthermore, most of these major publishers dedicate their resources for publishing content while consuming little to none content published by others, i.e., their level of content publication and consumption is very imbalanced. In addition to allocating a significant amount of resources for publishing content, these users often publish copyrighted material, which has legal implications for them [6] [7]. These observations raise the following question: *what are the main incentives of (major) content*

publishers in large BitTorrent portals?

To answer this important question, we conduct a systematic study on the major BitTorrent publishers. We show that these publishers can be broadly divided into two different groups: *top publishers* who publish a large number of often copyrighted content and *fake publishers* who publish a large number of fake content. We also identify the main characteristics (i.e., signature) of publishers in each group such as their seeding behaviour and the popularity of their published content. We investigate the main incentives of major (non-fake) publishers and classify them into the following three categories (or profiles): *(i) Private BitTorrent Portals* that offer certain services and receive financial gain through ads, donations and fees, *(ii) Promoting Web Sites* that leverage published content at BitTorrent portals to attract users to their own web site for financial gain, and *(iii) Altruistic Top Publishers*. We characterise these three groups of publishers and estimate the value (and income) of the associated web sites to support our claims about their incentives. Moreover, we are performing a detailed analysis of fake publishing phenomena. We conduct a separate large scale measurement study in order to analyse the presence of fake content in the BitTorrent ecosystem and to characterise fake publishers. We show that fake publishers can be divided into three groups: *(i) Malware Propagators* who spread the malware software *(ii) Scammers* who try to obtain financial benefits from BitTorrent users in a dishonest way and *(iii) Anti-piracy Agencies* who protect copyrighted content. Our results reveal that fake content represents an important portion (35%) of those files shared in BitTorrent and just a few tens of users are responsible for 90% of this content. Furthermore, more than 99% of the analysed fake files are linked to either malware or scam websites. This creates a serious threat for the BitTorrent ecosystem.

Finally, we define the notion of consumer loyalty towards a particular publisher and examine loyalty among BitTorrent consumers. We demonstrate that loyal consumers are mostly associated with top publishers. Furthermore, the fraction of loyal consumers and their level of loyalty towards a top publisher appear to be related to the publisher's profile as well as the type and the amount of published content.

The main contributions of this Chapter can be summarised as follows:

- We present a simple measurement methodology to monitor the content publishing activity in major BitTorrent portals. This methodology has been implemented in a system that continuously monitors and reports the content publishing activity in the Pirate Bay portal. The collected data by our system is made publicly available through our project web site [12].

- The distribution of the number of published content by each publisher is very skewed, i.e., a very small fraction of publishers (3%) is responsible for a significant fraction of the published content (67%) and for an even more significant fraction of download sessions (75%). These major publishers can be further divided into three groups based on their incentives as follows: *profit-driven* publishers, *altruistic top* publishers and *fake* publishers.

- *Profit-driven top* publishers own fairly profitable web sites. They use major BitTorrent portals such as the Pirate Bay as a platform to attract millions of users to their web site by showing the associated URL to a user at different steps of file download. The publishers that pursue this approach are responsible for roughly 30% of the content and 40% of the downloads in major BitTorrent portals.

- *Fake* publishers are either anti-piracy agencies or malicious users who are responsible for 30% up to 35% of the content and 25% of the downloads. These publishers sustain a continuous content poisoning attack [84] against major BitTorrent portals that affects millions of downloaders.

- We show that only the top publishers attract a significant number of loyal consumers. More interestingly, we observe that profit-driven publishers attract a larger absolute number of loyal consumers than altruistic top publishers (55K versus 6K) whereas altruistic publishers have a larger fraction of loyal consumers with a higher level of loyalty. This distinction appears to be directly related to their publishing strategy.

The rest of the Chapter is organised as follows. Section 5.2 describes our measurement methodology. Sections 5.3 and 5.4 are dedicated to the identification of major publishers and their main characteristics (i.e., signature) respectively. In Section 5.5, we study the key incentives for major content publishers. More detailed analysis of fake publishers can be found in Section 5.6. We examine consumer loyalty and its relationship with publisher profiles in Section 5.7. Section 5.8 presents other players that also benefit from content publishing. Finally Section 5.9 concludes the Chapter.

5.2 Measurement Methodology

The objective of our measurement study is to determine the identity of the initial publishers of a large number of torrents and to assess the popularity of each

	Portal	Start	End	#Torrents (username/IP)	#IP addresses
mn08	Mininova	09-Dec-08	16-Jan-09	- /20.8K	8.2M
pb09	Pirate Bay	28-Nov-09	18-Dec-09	23.2K/10.4K	52.9K
pb10	Pirate Bay	06-Apr-10	05-May-10	38.4K/14.6K	27.3M

Table 5.1: Datasets Description: portal name, start and end dates, number of collected torrents for which we identify the username and the IP address of the initial seeder, and number of collected consumers IP addresses

published file (i.e., the number and identity of peers who download the file).

Toward this end, we leverage the RSS feed to detect the availability of a new file on major BitTorrent portals and retrieve the publisher's username. In order to obtain the publisher's IP address, we immediately download the .torrent file and connect to the associated tracker. This implies that we often contact the tracker shortly after the birth of the associated swarm when the number of participating peers is likely to be small and includes the initial publisher (i.e., seeder). We retrieve the IP address of all participating peers as well as the current number of seeders in the swarm. If there is only one seeder in the swarm and the number of participating peers is not too large (i.e., < 20), we obtain the bitfield of available pieces at individual peers to identify the seeder. Otherwise, reliably identifying the initial seeder is difficult because either there is more than one seeder or the number of participating peers is large[1]. Furthermore, we cannot directly contact the initial seeder that is behind a NAT box and thus we are unable to identify the initial publisher's IP address in such cases. Using the above techniques we were able to reliably identify the publisher's username for all the torrents and the publisher's IP address in at least 40% of the torrents.

Once we identify a publisher, we periodically query the tracker in order to obtain the IP addresses of the participants in the associated swarm and always solicit the maximum number of IP addresses (i.e., 200) from the tracker. To avoid being blacklisted by the tracker, we issue our queries at the maximum rate allowed by the tracker (i.e., 1 query every 10 to 15 minutes depending on the tracker load). Given this constraint, we query the tracker from 8 geographically distributed machines[2]

[1]Our investigation revealed two interesting scenarios for which we could not identify the initial publisher's IP address: *(i)* swarms that have a large number of peers shortly after they are added to the portal. We discovered that these swarms have already been published in other portals. *(ii)* swarms for which the tracker did not report any seeder for a while or did not report a seeder at all.

[2]1 located in Oslo (Norway), 1 in Barcelona (Spain), 1 in Albacete (Spain) and 5 in different

so that all these machines collectively provide an adequately large probing rate to quickly discover most (and often all) of the participating peers and their evolution over time. We continue to monitor a target swarm until we receive 10 consecutive empty replies from the tracker. We use the MaxMind Database [18] to map all the IP addresses (for both publishers and downloaders) to their corresponding ISPs and geographical locations.

5.2.1 Dataset

Using the described methodology, we identify a large number of BitTorrent swarms at two major BitTorrent portals, namely Mininova and the Pirate Bay. Each one of these portals was the most popular BitTorrent portal at the time of the corresponding measurement according to Alexa ranking [9]. It is worth noting that the Pirate Bay is particularly interesting for our study since it is the only main BitTorrent portal where all the published content is contributed by users [118] (as opposed to being retrieved from other portals). Table 5.1 shows the main features of our three datasets (1 from Mininova and 2 from the Pirate Bay) including the start and end dates of our measurement, the number of torrents for which we identified the initial publisher (username/IP address), and the total number of discovered IP addresses associated for all the monitored swarms. We refer to these datasets as *mn08*, *pb09* and *pb10* throughout this Chapter. We note that dataset *mn08* does not contain the username of initial publishers and for dataset *pb09* we use a single query to identify initial publishers after detecting a new swarm through the RSS feed, but do not probe the tracker to capture all the consumers.

In Appendix B the potential biases of the measurement methodology are quantified.

5.3 Identifying Major Publishers

A publisher can be identified by its username and/or IP address. In our analysis, we identify individual publishers primarily by their username since the username is expected to remain consistent across different torrents. However, we require the identification of an individual publisher's IP address for network-level analyses such as determining the ISP where a publisher is located, the duration of time that a publisher remains in a swarm, or its participation across multiple swarms either

locations in Madrid (Spain).

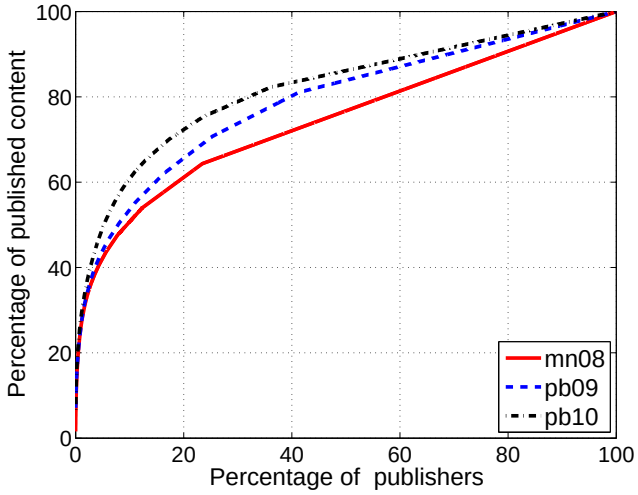


Figure 5.1: Percentage of content published by the top x% publishers.

as a publisher or a consumer. For these network-level analyses, we only consider those torrents in out dataset for which we are able to identify the IP address of the publisher (i.e., the initial seeder). Furthermore, we can identify publishers in *mn08* dataset only by their IP addresses since this dataset does not include the username of individual publishers. Finally, *fake* publishers are better identified by their IP addresses as we describe in the next Sections.

5.3.1 Skewness of Contribution

First, we examine the level of contribution (i.e., the number of published files) by the identified content publishers in each dataset. Figure 5.1 depicts the percentage of files that are published by the top x% of the publishers in our three datasets. We observe that the top 3% of the BitTorrent publishers contribute roughly 40% of the published content. Moreover, a more careful examination of IP addresses for the top-100 (i.e., 3%) publishers in our *pb10* dataset reveals that a significant fraction of them either do not download any content (40%) or download less than 5 files (80%). This large contribution of resources (bandwidth and content) by major publishers coupled with the significant imbalance between their publishing and consumption rates appears non-altruistic and rather difficult to justify for two simple reasons:

- Required Resources/Cost: publishing a large number of content requires a significant amount of resource (e.g., bandwidth). For example, a major content publisher named *eztv* recommends in its private BitTorrent portal web page (`www.eztv.it`) to allocate at least 10 Mbps in order to sustain the seeding of few (around 5) files in

mn08			pb09			pb10		
ISP	Type	%	ISP	Type	%	ISP	Type	%
OVH	Hosting Provider	13.31	OVH	Hosting Provider	24.76	OVH	Hosting Provider	15.16
Comcast	Commercial ISP	4.69	Comcast	Commercial ISP	3.67	SoftLayer Tech.	Hosting Provider	4.52
Keyweb	Hosting Provider	3.18	Road Runner	Commercial ISP	2.3	FDCservers	Hosting Provider	3.64
Road Runner	Commercial ISP	3.03	Romania DS	Commercial ISP	2.27	Open Computer Network	Commercial ISP	3.59
NetDirect	Hosting Provider	2.44	MTT Network	Commercial ISP	1.95	tzulo	Hosting Provider	3.36
Virgin Media	Commercial ISP	2.42	Verizon	Commercial ISP	1.64	Comcast	Commercial ISP	2.86
NetWork Operations Center	Hosting Provider	2.39	Virgin Media	Commercial ISP	1.49	Cosema	Commercial ISP	2.25
SBC	Commercial ISP	2.38	SBC	Commercial ISP	1.41	Telefonica	Commercial ISP	2.22
Comcor-TV	Commercial ISP	2.33	NIB	Commercial ISP	1.26	Jazz Telecom.	Commercial ISP	2.07
Telecom Italia	Commercial ISP	2.02	tzulo	Hosting Provider	1.14	4RWEB	Hosting Provider	2.06

Table 5.2: Content Publishers Distribution per ISP.

parallel.

- Legal Implications: As other studies have reported [38] and we confirm in our datasets, a large fraction of content published by major publishers is copyrighted material (recent movies or TV series). Thus, publishing these files is likely to have serious legal consequences for these publishers [6] [7].

This raises the question: *why do small fraction of publishers allocate a great deal of (costly) resources to contribute many files into BitTorrent swarms despite potential legal implications?* We answer this central question in Section 5.5.

5.3.2 Publishers' ISPs

To help identify content publishers in our dataset, we determine the ISP that hosts each major publisher and use that information to assess the type of service (and available resources) that a publisher is likely to have. Toward this end, we map the IP address for a publisher in each dataset to its corresponding ISP using the MaxMind database [18]. We then examine the publicly available information about each ISP (e.g., its web page) to determine whether it is a commercial ISP or a hosting provider. We perform this analysis only for the top-100 (roughly 3%) publishers since these publishers are mostly of interest and collecting the required information for all publishers is a tedious task. Since we do not have publishers' username for *mn08*, we examine the top-100 publishers based on their IP addresses in this dataset. For these publishers, we cannot assess the aggregated contribution of a publisher through different IP addresses (i.e., under-estimating the contribution of each publisher).

We observe that 42% of the top-100 publishers in *pb10*, 35% of the top-100 in *pb09* and 77% of the top-100 publishers in *mn08* are located at hosting services. Moreover 22%, 20% and 45% of these top-100 publishers are located at a particular hosting services, namely OVH, in *pb10*, *pb09* and *mn08* respectively.

	Published torrents	# IP addresses	# /16 IP Prefixes	# Geographical Locations
OVH (mn08)	2766	164	5	2
Comcast (mn08)	976	675	269	400
OVH (pb09)	2577	78	5	2
Comcast (pb09)	382	198	143	129
OVH (pb10)	2213	92	7	4
Comcast (pb10)	408	185	139	147

Table 5.3: Characteristics of all OVH and Comcast publishers in *mn08*, *pb09* and *pb10*.

In short, our analysis reveals that a significant fraction of major publishers are located at a few hosting services and a large percentage of them at OVH.

We also examine the contribution of BitTorrent publishers at the ISP-level by mapping all the publishers to their ISPs and identify the top-10 ISPs based on their aggregate published content for each dataset as shown in Table 5.2. This table confirms that content publishers who are located at a particular hosting provider, namely OVH, have consistently contributed to a significant fraction of published content at major BitTorrent portals. There are also several commercial ISPs (e.g., Comcast) in Table 5.2 with a much smaller contribution.

To assess the difference between users from hosting providers and commercial ISPs, we compare and contrast the characteristics of all publishers that are located at OVH and Comcast as representative ISPs for each class of publishers in Table 5.3. This table demonstrates the following two important differences: first, the aggregate contribution of each publisher at OVH is on average a few times larger than Comcast publishers. Second, Comcast publishers are sparsely scattered across many /16 IP prefixes and many geographical locations in the US whereas OVH publishers are concentrated in a few /16 IP prefixes and a handful of different locations in Europe (i.e., the location of OVH's data centres). In essence, the published content by Comcast publishers comes from a large number of typical altruistic users where each one publishes a small number of files likely from their home or work. In contrast, OVH publishers appear to be paying for a well-provisioned service to be able to publish a much larger number of files. We have also examined consuming peers (i.e., leechers) in captured torrents and did not observe the presence of OVH users among these consuming peers.

In summary, the examination of ISPs that host major BitTorrent publishers suggests that major publishers are located either at a few hosting providers (with a

large concentration at OVH) or at commercial ISPs. These publishers contribute a significantly larger number of files than average publishers. Furthermore, publishers who are located at hosting providers do not consume published content by other publishers.

5.3.3 A Closer Look at Major Publishers

We now examine the mapping between username and IP address of the top-100 content publishers in the *pb10* dataset to gain some insight about major publishers behaviour. Our examination reveals the following interesting points.

First, if we focus on the top-100 IP addresses that have published the largest number of files, only 55% of them are used by a unique username. The remaining 45% of the IP addresses of major publishers are mapped to a large number of usernames. We have carefully investigated this set of IP addresses and discovered that they use either hacked or manually created accounts (with a random username) to inject "fake" content. These publishers appear to be associated with anti-piracy agencies or malicious users. The former group tries to avoid the distribution of copyrighted content whereas the latter attempts to disseminate malware. We refer to these publishers as *fake publishers*. Surprisingly, fake publishers are responsible for around 25% of the usernames, 30% of the published content and 25% of the downloads in our *pb10* dataset. This suggests that major BitTorrent portals are suffering from a systematic poisoning index attack [84] that affects 30% of the published content. The portals fight this phenomenon by removing the fake content as well as the user accounts used to publish them. However, contrary to what has been reported in previous studies [99], this technique does not seem to be sufficiently effective since millions of users initiate the download of fake content. Finally, it is worth noting that most of the fake publishers perform their activity from three specific hosting providers named *tzulo*, *FDC Servers* and *4RWEB*. Due to the relevant activity of these fake publishers, we study them as a separate group in the rest of the Chapter. We also perform separate measurements and detailed analysis of fake publishers, which is presented in Section 5.6.

Second, the inspection of the top-100 usernames who publish the largest number of files shows that only 25% of them operate from a single IP. The remaining 75% of top usernames utilises multiple IPs and can be classified into the following common cases: *(i)* 34% of the usernames with multiple IP addresses (5.7 IP addresses on average) at a hosting provider in order to obtain the required resources for seeding a large number of files. (ii) 24% of the usernames with multiple IP addresses (13.8 IP

addresses on average) located at a single commercial ISP. Their mapping to multiple IP addresses could be due to the periodical change of their assigned IP addresses by their ISPs. (iii) The other 17% of these usernames are mapped to multiple IP addresses (7.7 IP addresses on average) at different commercial ISPs. These are users who inject content from various locations (for example, a user may publish from both home and work computer). To properly characterise different types of publishers, we exclude the 16 usernames who publish fake content from the top-100 usernames. We refer to the remaining top-100 usernames (non-fake publishers) as *Top* publishers who are responsible of 37% of the published content and 50% of the total downloads in our *pb10* dataset.

In summary, the major portion of the content comes from two reduced group of publishers: Top publishers and Fake publishers that collectively are responsible of 67% of the published content and 75% of the downloads. In the rest of the Chapter we devote our effort to characterise these two groups.

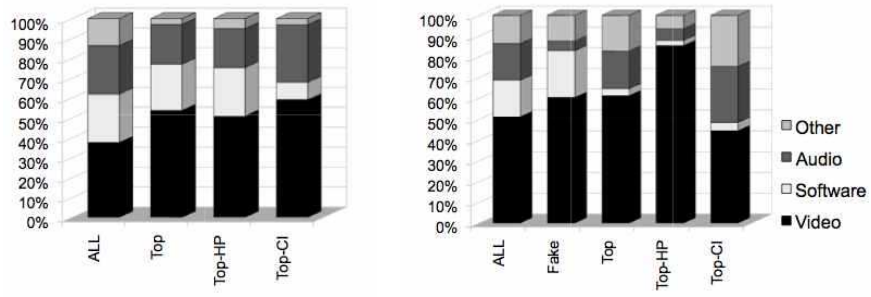
5.4 Signature of Major Publishers

Before we investigate the incentives of major BitTorrent publishers, we examine whether they exhibit any other distinguishing features, i.e., whether major publishers have a distinguishing signature. Any such distinguishing features could shed some light on the underlying incentives of these publishers. Toward this end, in the next few subsections, we examine the following characteristics of major publishers in our datasets: *(i)* the type of published content, *(ii)* the popularity of published content, and *(iii)* the availability and seeding behaviour of a publisher.

To identify distinguishing features, we examine the above characteristics across the following three *target groups* in each dataset: all publishers (labelled as "All"), all fake publishers (labelled as "Fake") and all top-100 (non-fake) publishers regardless of their ISPs (labelled as "Top"). We also examine the break down of Top publishers based on their ISPs into hosting providers and commercial ISPs, labelled as "Top-HP" and "Top-CI", respectively.

5.4.1 Content Type

We leverage the reported content type by each publisher to classify the published content across different target groups. Figure 5.2 depicts the breakdown of published content across different content type for all publishers in each target group for our



(a) *mn08* (b) *pb10*

Figure 5.2: Type of published content distribution for the different classes of publishers: *All*, *Fake*, *Top*, *Top-HP* and *Top-CI*.

Mininova and our major Pirate Bay datasets. We recall that without username information for each publisher in *mn08* dataset, we cannot identify fake publishers. Figure 5.2 reveals a few interesting trends.

First, Video files (which mainly include movies, TV-shows and porn content) constitute a significant fraction of published files across most groups with some important differences. The percentage of published Video across all publishers is around 37%-51% but it is slightly larger among top publishers. However, Video is clearly a larger fraction of published content by the top publishers located at hosting providers in our *pb10* dataset. Fake publishers primarily focus on Videos (recent movies and TV shows) and Software content. This supports our earlier observation that these publishers consist of anti-piracy agencies and malicious users because the former group publishes a fake version of recent movies while the latter provides software that contains malware.

5.4.2 Content Popularity

The number of published files by a publisher shows only one dimension of its contribution to BitTorrent. The other equally important issue is the popularity of each published content (i.e., the number of downloaders regardless of their download progress) by individual publishers. Figure 5.3 shows the box plot of the distribution of average number of downloaders per torrent per publisher across all publishers in each target group where each box presents the 25th, 50th and 75th percentiles. All the box plots presented in the rest of the Chapter follow the same format. On the one hand, the median popularity of top publishers' torrents is 7 times higher than a typical user (represented by *All*). A closer examination of the Top publishers

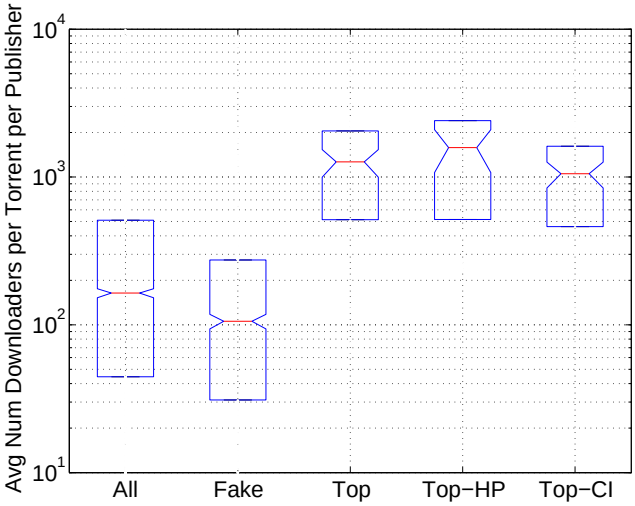


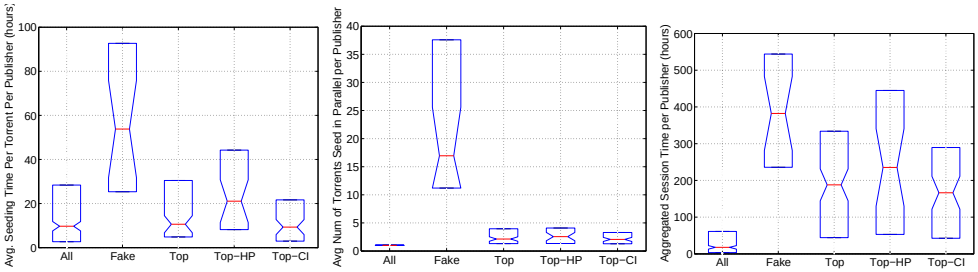
Figure 5.3: Avg. Num of Downloaders per torrent per publisher for the different classes of publishers: *All*, *Fake*, *Top*, *Top-HP*, *Top-CI*.

shows that the content published by users at hosting providers is on average 1.5 times more popular than those published by users at commercial ISPs. On the other hand, fake publishers' content is the most unpopular among the target groups. This is because the portals actively monitor the torrents and immediately remove the content identified as fake to avoid users from downloading it. Furthermore, users quickly realise the fake nature of these content and report this info on forums that inform others and limit their popularity.

In summary, top publishers are responsible for a larger fraction of popular torrents. This in turn magnifies the contribution of the 37% of the injected files by the top publishers to be responsible for 50% of all the downloads. The low popularity of fake publishers' content has the opposite effect and limits their contribution to the number of downloads to 25%.

5.4.3 Seeding Behaviour

We characterise the seeding behaviour of individual publishers in our target groups using the following metrics: *(i)* average seeding time of a publisher for its published content, (ii) average number of parallel seeded torrents, and (iii) aggregated session time of a publisher across all its torrents. Since calculating these properties requires detailed and computationally expensive analysis, we are unable to derive these values for all publishers. We use 400 randomly selected publishers to represent the normal behaviour of all publishers and refer to this group as "All"



(a) Average Seeding Time per Content per Publisher

(b) Average Number of Parallel Seed Torrents Per Publisher

(c) Average Aggregated Session Time Per Publisher

Figure 5.4: Seeding Behaviour for the different classes of publishers: *All*, *Fake*, *Top*, *Top-HP* and *Top-CI*.

in our analysis.

In order to compute these metrics we need to estimate the time that a specific publisher has been connected to a torrent (in one or multiple sessions). Since each query to the tracker just reports (at most) a random subset of 200 IPs, in big torrents (>200 peers), we need to perform multiple queries in order to assess the presence of the publisher in the torrent. In Appendix C, we detail the technique used to estimate the session time of a specific user in each torrent.

Average Seeding Time: We measure the duration of time that a publisher stays in a torrent since its birth to seed the content. In general, a publisher can leave the torrent once there is an adequate fraction of other seeds. Figure 5.4(a) depicts the summary distribution of average seeding time across all publishers in each target group. This figure demonstrates the following points: first, the seeding time for fake publishers is significantly longer than publishers in other groups. Since these publishers do not provide the actual content, the initial fake publisher remains as the only seed in the session (i.e., other users do not help in seeding fake content) to keep the torrent alive. Second, Figure 5.4(a) shows that top publishers typically seed a content for a few hours. However, the seeding time for top publishers from hosting providers is clearly longer than top publishers from commercial ISPs. This suggests that publishers at hosting providers are more concerned about the availability of their published content.

Average Number of Parallel Torrents: Figure 5.4(b) depicts the summary distribution of the average number of torrents that a publisher seeds in parallel across publishers in each target group. This figure indicates that fake publishers seed many torrents in parallel. We have seen that fake publishers typically publish a large num-

ber of torrents and other users do not help them for seeding. Therefore, fake publishers need to seed all of their seeded torrents in parallel in order to keep them alive. The results for top publishers show that their typical number of seeded torrents in parallel is the same (around 3 torrents) regardless of their location. However, we expect that a regular publisher seed only 1 file at a time.

Aggregated Session Time: We have also quantified the availability of individual publishers by estimating the aggregated session time that each publisher is present in the system across all published torrents. Figure 5.4(c) shows the distribution of this availability measure across publishers in each target group. As expected fake publishers present the longest aggregated session time due to their obligation to continuously seed their content to keep them alive. If we focus on top publishers, they exhibit a typical aggregated session 10 times longer than standard users. Furthermore, top publishers at hosting services are clearly more available than those from commercial ISPs.

5.4.4 Summary

BitTorrent content publishers can be broadly divided into three groups as follows: *(i)* Altruistic users who publish content while consuming content that is published by other users. *(ii)* Fake publishers publish a significant number of files that are often Video and Software content from a few hosting providers. Due to the fake nature of their content, their torrents are unpopular and they need to seed all torrents to keep them alive. These publishers appear to be associated with anti-piracy agencies or malicious users. We validate this hypothesis in Section 5.6. *(iii)* Top publishers publish a large number of popular files (often copyrighted video) and remain for a long time in the associated torrents to ensure proper seeding of their published content. These publishers are located at hosting facilities or commercial ISPs. Their behaviour suggests that these publishers are interested in the visibility of the published content possibly to attract a large number of users. The (cost of) allocated resources by these publishers along with legal implications of publishing copyrighted material cannot be considered as altruistic. Therefore, the most conceivable incentive for these publishers appears to be financial profit. We examine this hypothesis in the rest of the Chapter.

5.5 Incentives of Major Publishers

In this Section, we examine the incentives of different groups of publishers of non-fake content in more detail. Because of the relevance of the fake group of publishers, we conduct a separate measurement study and present the detailed analysis of those publishers in Section 5.6.

We are focusing on a group of major publishers who allocate significant amount of resources to publish non-fake an often copyrighted content. We believe that the behaviour of these users is not altruistic. More specifically, our hypothesis is that these publishers leverage major BitTorrent portals as a venue to freely attract consumers to their web sites. To verify this hypothesis, we conduct an investigation to gather the following information about each one of the *top* (i.e., top-100 non-fake) publishers:

- *Promoting URL*: the URL that downloaders of a published content may encounter,
- *Publisher's Username*: any publicly available information about the username that a major publisher uses in the Pirate Bay portal, and
- *Business Profile*: offered services (and choices) at the promoting URL.

Next, we describe our approach for collecting this information.

Promoting URL: We emulate the experience of a user by downloading a few randomly selected files published by each top publisher to determine whether and where they may encounter a promoting URL. We identified three places where publishers may embed a promoting URL: *(i)* the name of the downloaded file (e.g., user mois20 names his files as *filename-divxatope.com*, thus advertising the URL `www.divxatope.com`), *(ii)* the text-box in the web page associated with each published content, *(iii)* name of a text file that is distributed with the actual content and is displayed by the BitTorrent software when opening the .torrent file. Our investigation indicates that the second approach (using the text-box) is the most common technique among the publishers.

Publisher's Username: We browsed the Internet to learn more information about the username associated with each top publisher. First, the username is in some cases directly related to the URL (e.g., user UltraTorrents whose URL is `www.ultratorrents.com`). This exercise also reveals whether this username publishes on other major BitTorrent portals in addition to the Pirate Bay. Finally, posted

information in various forums could reveal (among other things) the promoting web site.

Business Services: We characterise the type of services offered at the promoting URL and ways that the web site may generate income (e.g., posting ads). We also capture the exchanged HTTP headers between a web browser and the promoting URL to identify any established connection to third-party web sites (e.g., redirection to ads web sites or some third party aggregator) using the technique described in [74].

5.5.1 Classifying Publishers

Using the above methodology, we examined a few published torrents for each one of the top publishers as well as sample torrents for 100 randomly selected publishers that are not in the top-100, called *regular publishers*. On the one hand, we did not discover any interesting or unusual behaviour in torrents published by regular publishers and thus conclude that they behave in an altruistic manner. On the other hand, a large fraction of seeded torrents by the top publishers systematically promotes one or more web sites with financial incentives. Our examination revealed that these publishers often include a promotional URL in the text-box of the content web page. We classify these top publishers into the following three groups based on their type of business (using the content of their promoting web sites) and describe how they leverage BitTorrent portals to intercept and redirect users to their web sites.

Private BitTorrent Trackers: A subset of major publishers (25% of top) owns their BitTorrent portals that are in some cases associated with private trackers [63]. These private trackers offer a better user experience in terms of download rate (compared to major open BitTorrent portals) but require clients to maintain a certain seeding ratio. More specifically, each participating BitTorrent client is required to seed content proportionally to the amount of data it downloads across multiple torrents. To achieve this goal, users are required to register in the web site and login before downloading the torrent files. The publishers in this class publish 18% of all the content while they are responsible for 29% of the downloads. 2/3 of these publishers advertise the URL in the text-box at the content web page. Furthermore, they appear to gain financial profit in three different ways: (i) posting advertisement in their web sites, (ii) seeking donations from visitors to continue their basic service, and (iii) collecting a fee for VIP access that allows the client to download any content without requiring any kind of seeding ratio. These publishers typically inject video, audio and software content into BitTorrent portals. Interestingly, a significant fraction of

publishers in this class (40%) publish content in a specific language (Italian, Dutch, Spanish or Swedish) and specifically a 66% of this group are dedicated to publishing Spanish content. This finding is consistent with prior reports on the high level of copyright infringement in Spain [28].

Promoting Web Sites: Another class of top publishers (23% of top) appears to be promoting some URLs that are associated with hosting images web sites (e.g., `www.pixsor.com`), forums or even religious groups (e.g., `lightmiddleway.com`). These publishers inject 8% of the content and are responsible of 11% of the downloads. Most publishers in this class advertise their URL using the text-box in the content web page. Furthermore, most of these publishers (70%), specifically those that are running a hosting image web site, publish only porn content. Inspection of the associated hosting image web sites revealed that they store adult pictures. Therefore, by publishing porn content in major BitTorrent portals, they are targeting a particular demography of users who are likely to be interested in their web sites. The income of the portals within this class is based on advertisement.

Altruistic Publisher: The remaining top publishers (52% of top) appear to be altruistic users since they do not seem to directly promote any URL. These publishers are responsible of 11.5% of the content and roughly the same fraction of downloads. Many of these users publish small music and e-book files that require smaller amount of seeding resources. Furthermore, they typically include a very extensive description of the content and often ask other users to help with seeding the content. These evidences suggest that these publishers may have limited resources and thus they need the help of others to sustain the distribution of their content.

In summary, roughly half of the top publishers advertise a web portal in their published torrents. It appears that their intention is to attract a large number of users to their web sites. The income of these portals comes from ads and in the specific case of private BitTorrent portals also from donations and VIP fees. Overall, these profit-driven publishers generate 26% of the content and 40% of the downloads. Therefore, the removal of this small fraction of publishers may have a significant impact on the popularity of major BitTorrent portals. Finally, a fraction of publishers appears to be altruistic and responsible for a notable fraction of published content and downloads (11.5%). This suggests that there are some seemingly ordinary users who dedicate their resources to share content with a large number of peers in spite of the potential legal implications of such activity.

	Lifetime (days)	Avg. Publishing Rate (torrents per day)
Private Portals	63/466/1816	0.57/11.43/79.91
Promoting Web sites	50/459/1989	0.38/4.31/18.98
Altruistic	10/376/1899	0.10/3.80/23.67

Table 5.4: Lifetime and Avg. Publishing Rate for the different classes of content publishers: BitTorrent Portals, Promoting Web Sites and Altruistic Publishers. The represented values are min/avg/max per class.

5.5.2 Longitudinal View of Major Publishers

So far, we focused on the contribution of major publishers only during our measurement intervals. Having identified the top publishers in our *pb10* dataset, we examine the longitudinal view of the contribution by major publishers since they appeared on the Pirate Bay portal. Toward this end, for each top publisher, we obtain the username page on the Pirate Bay portal that maintains the information about all the published content and its published time by the corresponding user till our measurement date (June 4, 2010)[3]. Using this information for all top publishers, we capture their publishing pattern over time with the following parameters: *(i) Publisher Lifetime* which represents the number of days between the first and the last appearance of the publisher in the Pirate Bay portal, *(ii) Average Publishing Rate* that indicates the average number of published content per day during their lifetime.

Table 5.4 shows the min/avg/max value of these metrics for the different classes of publishers: Private Portals, Promoting Web Sites and Altruistic publishers. The profit-driven publishers (i.e., private portals and promoting web sites) have been publishing content for 15 months on average (at the time of the measurement) while the most longed-lived ones have been feeding content for more than 5 years. Furthermore, some of these publishers exhibit a surprisingly high average rate of publishing content (80 files per day). The altruistic publishers present a shorter lifetime and a lower publishing rate that seems to be due to their weaker incentives and their lower availability of resources.

In summary, the lifetime of major publishers suggests that content publishing in BitTorrent seems to have been a profitable business for (at least) a couple of years. Furthermore, the high seeding activity by profit-driven publishers (e.g., private portals) over a long period of time implies a high and continuous investment for re-

[3]Note that we cannot collect information about fake publishers since the web pages of their associated usernames are removed by the Pirate Bay as soon as they are identified.

	Web site Value ($)	Web site Daily Income ($)	Web site Daily visits
Private Portals	1K/33K/313K/2.8M	1/55/440/3.7K	74/21K/174K/1.4M
Promoting Web Sites	24/22K/142K/1.8M	1/51/205/1.9K	7/22K/73.5K/772K

Table 5.5: Publisher's web site value ($), daily income ($) and num of daily visits for the different classes of profit-driven content publishers: BitTorrent Portals and Promoting Web Sites. The represented values are min/median/avg/max per class.

quired resources that should be compensated by different types of income (e.g., ads) for these portals. We examine the income of the profit-driven publishers in the next subsection.

5.5.3 Estimating Publishers' Income

The evidences we presented in previous subsections suggest that the goal of half of the top publishers is to attract users to their own web sites. We also showed that most of these publishers promote conditions to generate income by posting ads in their web sites. In essence, these publishers have a clear financial incentive to publish content. In order to validate this key point, we assess their ability to generate income by estimating three important but related properties of their promoting web sites: *(i)* average value of the web site, *(ii)* average daily income of the web site, and *(iii)* average daily visits to the web site. In the absence of a reliable source to obtain this sensitive information, we rely on several companies[4] that monitor and report these statistics for different web sites. Since these companies do not reveal the details of their monitoring strategy, we cannot assess the accuracy of their reported statistics. To reduce any potential error in the provided statistics by individual companies, we collect this information from six independent companies and use the average value of these statistics among them. We emphasise that the obtained statistics from these companies are treated as ballpark estimates for the three properties of the promoting web sites to enable our validation.

Table 5.5 presents the min/median/avg/max value of the described metrics for

[4] `sitelogr.com`, `cwire.com`, `websiteoutlook.com`, `sitevaluecalculator.com`, `mywebsiteworth.com`, `yourwebsitevalue.com`

each class of profit-driven publisher classes (i.e., private portals and promoting web sites). The median values suggest that the promoted web sites are fairly profitable since they value tens of thousand dollars with daily income of a few hundred dollars and tens of thousand visits per day. Furthermore, few publishers ($<$10) are associated to very profitable web sites, valued in hundreds of thousand to millions of dollars, which receive daily income of thousands of dollars and hundreds of thousand visits per day.

In summary, these statistics suggest that these web sites are valuable and visible, and generate a substantial level of income.

5.6 Fake Content in BitTorrent Ecosystem

In this Section, we thoroughly analyse the *fake publishing* phenomenon in BitTorrent in order to understand its real impact on the system performance as well as the potential risks of fake content for BitTorrent users. We base our study on additional dataset which we collected from torrents published in the Pirate Bay portal during a period of 14 days from 30-04-2011 to 13-05-2011. The 35% of almost 30K analysed torrents are associated to fake content. This depicts a 5% increment in the presence of fake content within the BitTorrent ecosystem comparing to the previous measurements (*pb10* dataset) in a period of one year. This justifies (even more) the necessity of the research conducted in this Section.

In order to fight the fake publishing phenomenon, the first step is to properly characterise the fake publishers and their behaviour. The current BitTorrent portals solutions identify fake publishers through the user account that they use to upload fake torrents to the portal. We show that this technique is inefficient since the fake publisher can generate as many user accounts as needed in those portals. Instead, the parameter that uniquely identifies the fake publisher is the IP address it uses to perform its activity. Surprisingly, our data reveals that just 20 fake publishers (whose IP we identify) are responsible for injecting 90% of fake content in a major BitTorrent portal. Moreover, most of these IP addresses belong to Hosting Providers where the fake publishers rent dedicated high-resource servers to perform their activity.

Therefore, the fake publishing activity is time consuming since a fake publisher needs to manually create the user accounts used in the different portals (in some cases up to 4 accounts per day). Furthermore, this activity requires dedicated resources (e.g., rented servers). This investment in time and resources can be only justified by a strong motivation behind the distribution of fake content. We have

downloaded and manually inspected a large number of fake content published during our measurement period and found 3 different profiles among the fake publishers: (i) a first group of fake publishers aims to spread malware using the popular BitTorrent system; (ii) a second set of users tries to attract BitTorrent users to scam websites in order to get economical benefit from the victims by using different scam techniques; (iii) the last group is formed by anti-piracy agencies that upload fake versions of those content that they want to protect.

Our data shows that more than 99% of the published fake content is associated with the two first profiles. This supposes a very serious threat for the BitTorrent ecosystem since the activity of these publishers may lead to thousands of undesirable episodes of scammed users and computer infections. These findings suggest that new solutions need to be proposed in order to eliminate or at least reduce the number of fake content available in the BitTorrent ecosystem. Towards this end, we have designed and implemented TorrentGuard. This is a novel detection tool that allows to identify the IP address of the fake publisher, thus being able to report as fake each content published from this IP address at the moment of its publication. Based on the performed evaluation, TorrentGuard would be able to avoid more than 35 millions fake content downloads every year. This means, preventing hundreds of thousands of users to suffer from computer infections or scam incidents every year. The detailed description and evaluation of TorrentGuard solution is presented in Section 6.3

Next, in Section 5.6.1 we describe our measurement methodology and present our dataset. Section 5.6.2 characterises fake publishers, while Section 5.6.3 classifies them depending on the goal they pursuit with their activity. Section 5.6.4 shortly characterises the downloaders of the fake content.

5.6.1 Measurement of Fake Publishers

In this Section we specifically address the fake content publishing phenomenon in BitTorrent. A fake publisher is a user that exploits the BitTorrent ecosystem to publish fake content, this is, content that is different from what is expected from the content name. A fake publisher makes available the fake content from a single IP address (or limited number of IP addresses) that corresponds to the initial seeder of all its published content. Furthermore, a fake publisher typically creates a user account in a BitTorrent portal from which it uploads .torrent files associated with its fake content. Some portals, such as the Pirate Bay, removes this user account after some client reports that it is being used to publish fake content. Then, the fake publisher reacts by creating a new account to publish new .torrent files and this

loop keeps repeating. Hence, contrary to the case of regular publishers (that can be identified by its associated username in the BitTorrent portal), fake publishers can exclusively be identified by its IP address. Finally, it must be noted that, to the best of the authors' knowledge, the previously described technique based on users' reports is the only one used nowadays for detecting and deleting fake content.

5.6.1.1 Measurement Methodology

The main objective of this measurement study is to identify fake publishers. This subsection aims to describe the methodology we followed to achieve the main goals of this Section: identify the IP addresses of the fake publishers, capture the time the Pirate Bay portal takes before removing fake publishers accounts from its site and estimate the number of the downloads of fake contents that take place until (and after) the fake user account is deleted. In order to understand such methodology we first describe how BitTorrent system operates and we present the definition of a fake publishers and their behaviour in the system.

Towards this end, our measurement tool has three independent modules. The first one is responsible for finding the IP address and username of the publisher associated with each announced content in the Pirate Bay. This method is the same as described in Section 5.2. Therefore, using the described methodology we are able to characterise the content publisher by both its username and IP address in many cases.

The second module of our tool is responsible for identifying those publishers that are in fact fake ones. For this purpose our tool connects periodically (every 5 minutes) to the Pirate Bay web page of each known publisher. If at some point the Pirate Bay web page has been removed we consider that the IP address associated with the removed account belongs to a fake publisher. Furthermore, we also collect the time that the Pirate Bay requires to detect and eliminate each fake publisher account.

Finally, our tool has a third module that counts the number of peers that connect to the swarm of each fake content in order to download it. Specifically, our tool systematically queries the tracker(s) managing the download of each fake content to obtain those IP addresses participating in the swarm. In order to accelerate this process we perform this task from four independent machines.

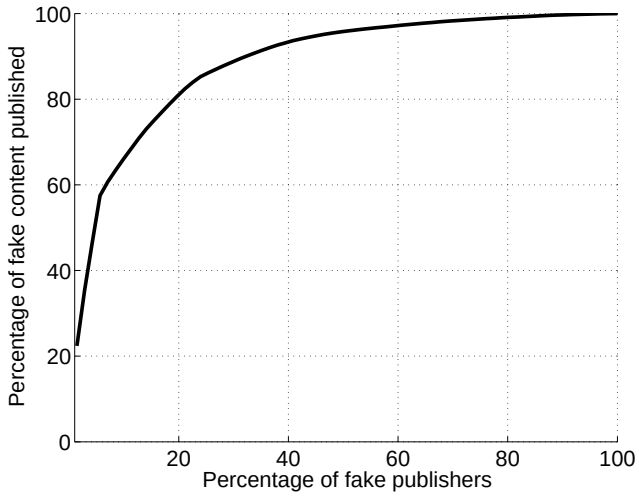


Figure 5.5: Percentage of fake content published by the top x% fake publishers

5.6.1.2 Dataset Description

We have applied the described methodology between 30-04-2011 and 13-05-2011, in addition to 5 days of warm-up phase dedicated to identify the initial fake publishers' IP addresses. During the measurement period we have collected 29330 torrents, from which 10206 (35%) were identified as fake ones. Furthermore, we have collected the IP addresses of those peer participating in swarms associated with fake content until two instants: (i) the moment the content is removed from the Pirate Bay and (ii) the end of our measurement study.

5.6.2 Fake Publishers Characterisation

Our results reveal that more than 1/3 of the content published in the Pirate Bay is fake. This shows an increasing trend in the number of fake content comparing to *pb10* dataset when the fake content represented 30%. Therefore, it is critical to eliminate or at least reduce this huge number of fake content in BitTorrent ecosystem. The first step towards this end is identifying who is responsible for publishing this fake content and characterising their behaviour. In this Section, we address this issue using the collected data. More specifically, we aim to answer questions such as: *How many fake publishers (i.e., IP addresses) are uploading fake content to the BitTorrent Ecosystem? From where (i.e., which ISP) they perform their activity? How frequently they upload fake content?*

5.6.2.1 Number and Contribution of Fake Publishers

Unexpectedly, we observe that only 71 IP addresses are responsible for those 4779 fake content for which we identified the initial seeder. This implies almost 70 fake content published from each of these IPs in average. However, it is interesting to investigate the level of the contribution of each one of these fake publishers. Towards this end, Figure 5.5 depicts the percentage of fake content published by the top x% of these fake publishers. The graph shows a skewed distribution where 10 IPs (14%) are responsible for publishing almost 75% of all the fake contents. Moreover, this number increases to 90% if we consider the top 20 IP addresses. Therefore, we can conclude that a reduced number of just 20 fake publishers are responsible for poisoning the BitTorrent ecosystem. In the rest of the Section we focus on thoroughly studying this group of 20 fake publishers that we refer to as *Top Fake Publishers*.

5.6.2.2 Location of Fake Publishers

We have mapped the IP address of each one of the Top Fake Publishers to its correspondent ISP using the MaxMind database [18]. Surprisingly, 17 out of the Top 20 fake publishers operate from Hosting Providers. These are companies dedicated to rent high-resources (CPU, memory and bandwidth) provisioned servers. Moreover, 70% of the fake content is seeded from just two Hosting Providers named *OVH Systems* and *Obtrix* located at France and New Zealand respectively.

Fake publishers need on the one side resources in order to sustain the distribution of a large number of fake files [48] and on the other side anonymity due to the *illegitimate* activity being performed. The usage of rented servers in Hosting Providers covers both requirements.

Hence, the use of dedicated servers in Hosting Providers reveals that most of the fake publishers perform their activity from a stable IP since those servers typically have a static IP address configured. This makes them easily identifiable. In this sense, the usage of anonymity services such as TOR [25] or proxy services seems to be useful for fake publishers in order to difficult their identification. However, we have not found that the fake publishers identified in our dataset use such services. This suggests that the severe performance degradation associated to these anonymity services prevent the fake publishers from using them.

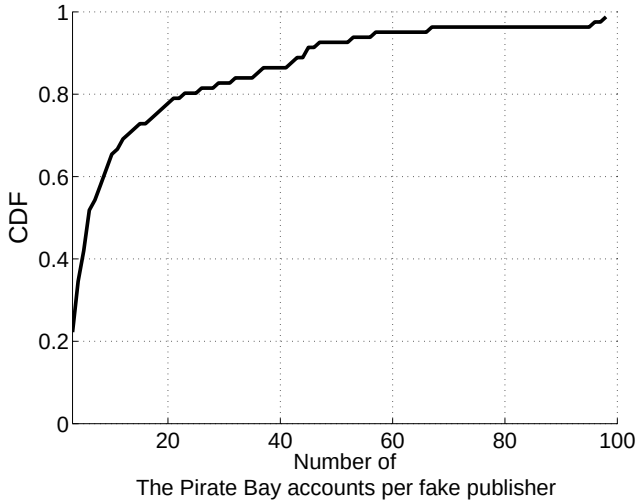


Figure 5.6: CDF of the number of the Pirate Bay accounts per fake publisher

5.6.2.3 Pirate Bay Accounts Utilisation

The Pirate Bay solicits to solve a CAPTCHA [13] in order to create an account to avoid the automatic generation of accounts. Hence, fake publishers are obeyed to create their accounts manually. Figure 5.6 shows the CDF of the number of the Pirate Bay accounts used by each one of the 71 identified fake publishers. A fake publisher uses (in median) 6 accounts in a period of 14 days. However, a 5% of the fake publishers inject content using more than 58 different accounts in the same period. This represents an average number of 4 accounts per day. This result suggests that fake publishers need to dedicate time to track the availability of their accounts in order to manually generate new ones if needed.

Interestingly, we also observe a second strategy that although marginal is worth to report. In these cases, fake publishers hijack the accounts with a legitimate publishing history. This provides a trust reputation among the downloaders. Therefore, this could extend the time that fake user could be injecting fake torrents before being reported. However, due to the required technical skills for applying this technique, this case represents less than 1% of all fake accounts.

5.6.2.4 Publishing Strategies

Fake users follow two different strategies to upload fake contents into the Pirate Bay portal. On the one hand, we found users that publish a large number of fake content in a row (typically around 10) in just few seconds after creating a user ac-

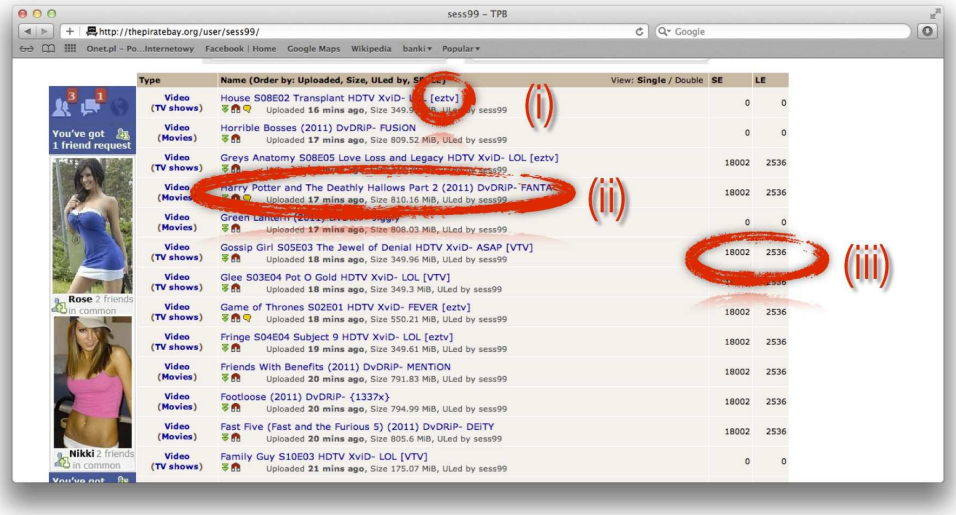


Figure 5.7: The example of Pirate Bay web page of malicious user: *(i)* the publisher creates the impression to be a trusted user (eztv) *(ii)* the publisher uses a popular content title *(iii)* the publisher falsifies statistics about number of seeders and leechers

count. Once the account is deleted, they repeat the same process from a new account. Around 70% of Top Fake Publishers use this technique. On the other hand, 30% of the Top Fake Publishers upload just one or two fake contents with a username. This is a more conservative technique that extends the time that those fake accounts are active before being eliminated when compared to the previous case. Specifically, the accounts of those publishers using the first strategy are detected and then deleted in 92 minutes (in average) whereas the accounts of those using the second strategy are deleted in 253 minutes, thus being their content available 2.75 times more time in the Pirate Bay. Unexpectedly, although the second strategy offers longer accounts' lifetime, it attracts only 47 downloaders per torrent (in average) in front of the 84 attracted by fake publishers using the first strategy. This happens because the fake publishers using the first strategy typically use popular names associated to their content whereas publishers using the second more conservative strategy do not use such popular names.

5.6.2.5 Strategies to Attract Downloaders

The main goal of fake publishers in BitTorrent is to produce as many downloads of their content as possible. Therefore, they need to offer torrents that sound very

attractive for the downloaders. Towards this end, we have observed that fake publishers use three different strategies: *(i)* they assign to the content a very popular name such as the title of the last released Hollywood movies; *(ii)* creating the false impression that the content has been published by a well-known and trusted user. For this purpose, the fake publisher names its content in the same way as the trusted one. For instance, eztv one of the most popular publisher in the Pirate Bay adds the signature [eztv] at the end of the title of its published files. Then, some fake publishers also add this signature to the title of their fake content; *(iii)* presenting attractive performance statistics (i.e., a high number of seeders and leechers) for the fake torrent. In this way, the fake torrent is perceived as a very popular torrent by the downloaders, which assume they will obtain a high download rate in case of selecting that torrent. In order to generate these fake statistics the publisher connects to the Tracker many times using a single IP but different ports. Then the tracker considers each one of these IP+port pairs as a single peer and reports a high number of seeders and leechers. The Pirate Bay retrieves and presents these statistics from the Tracker. All of those strategies are summarised in Figure 5.7.

In summary, the fake content publishing activity is performed from Hosting Providers facilities by just few dozens of users. Furthermore, fake publishers are aware of how the BitTorrent ecosystem works, thus they use sophisticated strategies in order to improve the success of their activity.

5.6.3 Fake Publishers Profiles

After characterising the fake publishers behaviour, we still need to answer an important question: *What incentives a user has to publish fake content?* To answer this question we have downloaded up to 10 files published by each fake publishers in our dataset and manually inspected them. Our analysis reveals the presence of three different profiles: malware propagators, scammers and anti-piracy agencies. Next, we describe in detail each one of these profiles.

5.6.3.1 Malware Propagators

These users exploit the popularity of BitTorrent system in order to rapidly propagate malware among thousands of users. On the one hand, for some of the users in this group the published content is the malware itself. In this case, the content including the malware pretends to be typically a patch for a popular game, a key generator, etc. On the other hand, a second set of users use a more sophisticated

technique. They publish a movie with a catchy title. The content has the standard size of a DivX movie (i.e., between 700 MB and 1 GB), and even sometimes includes a second small file with a real sample of the movie. Hence, the file has the appearance of a non-fake legitimate content. However, when a user downloads the content and tries to play the movie, it is requested to reproduce it using Windows Media Player (WMP) in case a different player is run instead. When the movie is finally reproduced with the WMP a pop-up window appears requesting to install new codecs along with a url link from where these codecs can be downloaded. Of course, the file including those pretended codecs is reported as a malware by security and anti-virus software.

5.6.3.2 Scammers

In this case, the fake publisher uses a similar technique to the sophisticated one described above. However, when the user plays the movie with WMP, it is automatically redirected to a website in the Internet. A second variant used by scammers is to provide a file protected with a password (typically .rar), and offer the user a website in which the password can be obtained. Once the user gets into one of these websites, a credit card payment is requested in order to obtain some privilege to watch the downloaded movie (e.g., the password of the .rar file). In some other situations the user is informed that in order to check he is not a bot, a survey must be filled previously to watch the movie. This survey results to be a contest in which you are obeyed to subscribe to a paid premium SMS service. These websites are often reported as scam on different forums, one example of them is http://movieyt.com.

We have performed a more detailed analysis of these websites. On the one hand, when a user wants to abandon the web page several pop-up windows appear trying to change user mind and making leaving the web page at least bothersome. On the other hand, when a user enters some of these web pages, a pop-up window advertising a Facebook group of the web page shows up. This pop-up does not react to the explorer close button, rather, just by clicking on the "I like it" Facebook button the window closes. This method aims to increase the trust of the web page so that users interpret it is a legitimate website. More importantly, this finding suggests that these scammers do not limit their activity to BitTorrent but they also try to capture victims from other popular applications such as online social networks.

Country	Percentage of BitTorrent users downloading fake content	Percentage of BitTorrent users	The ratio
United States	12.40%	10.42%	1.19
China	6.27%	4.20%	1.49
Great Britain	4.60%	6.26%	0.73
Brazil	4.26%	2.68%	1.59
Italy	3.88%	4.13%	0.94
India	3.78%	5.71%	0.66
Canada	3.29%	3.85%	0.85
Spain	2.79%	5.95%	0.47
Austria	2.73%	2.83%	0.96
Poland	2.66%	2.86%	0.93

Table 5.6: Demographics of BitTorrent users vs. fake content downloaders per country (the third column represent the ratio column 1/column 2)

5.6.3.3 Anti-piracy Agencies

The two previous profiles have dishonest purposes. Anti-piracy agencies instead, publish fake version of the copyrighted content that they want to protect. This content however, is not what downloader is expecting from the title (e.g., copyrighted movie). Sometimes this content includes anti-piracy adverts. The action performed by anti-piracy agencies is limited in the number of contents (under request from a company) and time (in the weeks before and after the content, e.g., movie, is released).

In summary, we distinguish three different profiles among fake publishers that motivates them to perform their activity. On the one hand, 65% of the Top Fake Publishers in our dataset are malware propagators and are responsible for around a 30% of the published fake content. On the other hand, a 35% of the Top Publishers are scammers and they published a 70% of the fake content during our measurement period. Finally, anti-piracy agencies represent a very small fraction of the fake content published due to the specificity of their actions.

In conclusion, it is worth to mention that the content published by malware propagators and scammers is potentially harmful, especially for not technically skilled downloaders. Hence, they represent a serious risk for the BitTorrent ecosystem that should be erased or at least mitigated. We address this issue in Section 6.3.

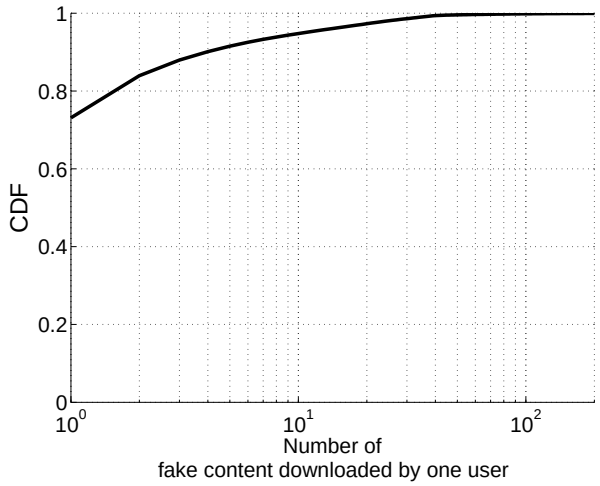


Figure 5.8: CDF of the number of fake content downloaded by one user

5.6.4 Characterising the Downloaders of Fake Content

In this Section we look at the studied phenomenon from the victims side. First, we analyse the demographics of the victims and group them per country in order to understand which countries suffer more from the reported problem. In order to provide full meaningful results we have compared the demographic distribution of the victims of fake content with the demographic of distribution of BitTorrent clients obtained from the *pb10* dataset.

Table 5.6 offers the obtained results. It shows the percentage of victim downloaders of fake content, the percentage of BitTorrent users and the ratio between these two percentages for the 10 countries with a larger number of victims. If the victims were randomly selected, this ratio would be close to 1. However, this is not the case. On the one hand, we observe that some countries such as US, China and Brazil shows a ratio > 1. For instance, Brazil has a ratio equal to 1.59. This means that Brazil has 59% more victims than expected from a random process. On the other hand, countries such as UK, India or Spain shows a value < 1. For instance Spain has a ratio equal to 0.47. This means that Spain only has 47% of the victims it should have from a random process.

Next, we study the number of fake content downloads performed by a single user. This help to understand whether there are users that are highly vulnerable to the described threats. Figure 5.8 shows the CDF of the number of fake content downloaded by each victim. We can see that 70% of the victims downloaded just 1 fake content. However, it is worth to note the presence of users who downloaded

multiple fake torrents during the measurement period.

The obtained results suggest that users from some specific countries (those having a ratio less than 1) are more skilled to identify fake content so being more protected against possible infections and/or scam episodes.

5.7 Examining Consumer Loyalty

In this Section, we examine the behaviour of consumers towards individual top publishers and their relationship with publishers' profile. More specifically, our goal is to answer two basic questions:

- Can individual top publishers attract loyal consumers that primarily download content from that top publisher?
- Is there any correlation between a publisher's profile and the level of loyalty among its consumers?

In essence, answering these questions reveals whether top publishers adopt business practices that affect their ability to form a loyal consumer-base within the BitTorrent ecosystem in order to achieve their goals. The notation used in this Section is shown in Table 5.7.

5.7.1 Quantifying Consumer Loyalty

To study the loyalty of consumers, we need to define a meaningful metric to quantify this attribute of a consumer towards a particular publisher. Suppose that consumer c downloads $dl(c)$ files from $pub(c)$ different publishers where the largest number of downloaded files by c from a publisher is $C(c)$ ($C(c) \leq dl(c)$). Consumer

Metric	Definition
pub(c)	Num. of publishers from which consumer c downloads files
dl(c)	Num. of files downloaded by consumer c
C(c)	Num. of files downloaded by consumer c from its preferred publisher
L(c)	Normalised loyalty of consumer c to its preferred publisher
NLC(p)	Absolute number of loyal consumers for publisher p
FLC(p)	Fraction of loyal consumers for publisher p

Table 5.7: Notation used in Section 5.7 (Examining Consumer Loyalty)

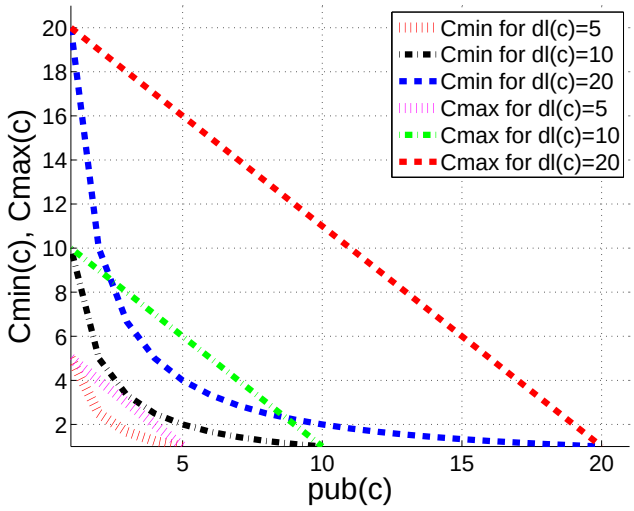


Figure 5.9: Max and Min value for $C(c)$ for different $pub(c)$ and $dl(c)$ values.

c is considered loyal to publisher p if it downloads most of its files from p. We refer to p as the *preferred* publisher for consumer c.

On the one hand, if the downloaded files by consumer c are evenly divided among $pub(c)$ publishers, c is *not* loyal to any publisher since it shows the minimum consumption level ($C_{min}(c)$) towards any publisher that is simply:

$$C_{min}(c) = \frac{dl(c)}{pub(c)} \tag{5.1}$$

On the other hand, for a given consumer c, the consumption from its preferred publisher is maximised when c downloads only one file from each non-preferred publisher and all remaining files from its preferred publisher. Thus $C_{max}(c)$ can be easily expressed as:

$$C_{max}(c) = dl(c) - pub(c) + 1 \tag{5.2}$$

Given a particular scenario defined by $dl(c)$ and $pub(c)$, the above simple equations for $C_{min}(c)$ and $C_{max}(c)$ determine the possible range for the number of files that a user c can download from a publisher. Figure 5.9 shows the variation of $C_{max}(c)$ and $C_{min}(c)$ as a function of $pub(c)$ for different $dl(c)$. This figure reveals that both $C_{max}(c)$ and $C_{min}(c)$ can significantly change across different scenarios as $dl(c)$ and $pub(c)$ vary. To allow comparison of the loyalty of users across different scenarios, we define the loyalty of user c towards his preferred publisher as his normalised consumption as follows:

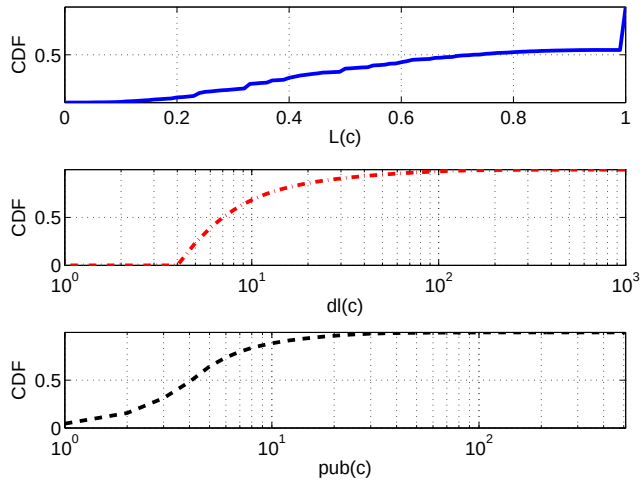


Figure 5.10: Distribution of $L(c)$, $dl(c)$ and $pub(c)$ across all consumers with $dl(c) \geq 5$

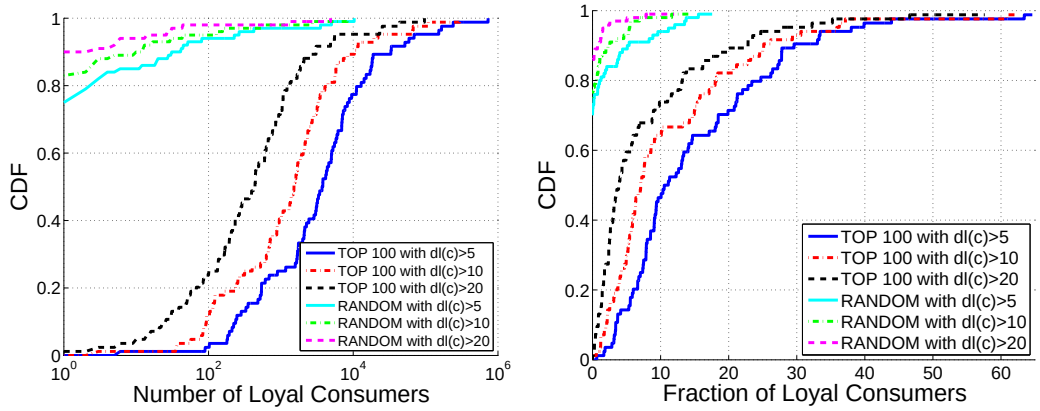
$$L(c) = \frac{C(c) - C_{min}(c)}{C_{max}(c) - C_{min}(c)} \text{ so that } 0 \leq L(c) \leq 1 \tag{5.3}$$

We use our *pb10* dataset for this analysis. This dataset contains 27.3M consumers, however, we only focus on 2.6M consumers that are moderately active (i.e., have downloaded at least five files, $dl(c) \geq 5$) and exhibit a positive loyalty ($L(c) > 0$). Figure 5.10 shows the distribution of $pub(c)$, $dl(c)$ and $L(c)$ among these consumers. On the one hand, we observe that a majority (85%) of these consumers download less than 19 files and from less than 8 different publishers during our one-month measurement period. This figure shows that a roughly 45% of these consumers exhibit $L(c)$ value very close to 1 and the median $L(c)$ value is 0.73. This suggests that half of these consumers exhibit a rather high level of loyalty towards a particular publisher. For the rest of analysis in this section, we focus on all moderately active consumers with positive loyalty and their corresponding publishers. We also filter consumers based on their $dl(c)$ values to examine more active consumers.

5.7.2 Consumer Loyalty Among Publishers

We first examine the level of loyalty among consumers of two group of publishers (as our target groups) as follow:

- *(i) top-100*: the non-fake top-100 publishers that we identified in Section 5.3. We recall that only 84 of the top-100 publishers were non-fake.



(a) NLC (b) FLC

Figure 5.11: CDF of $NLC(p)$ and $FLC(p)$ for non-fake top-100 publishers

- *(ii) Random*: 100 non-fake randomly selected publishers (excluding the top-100) to represent the rest of publishers in our dataset.

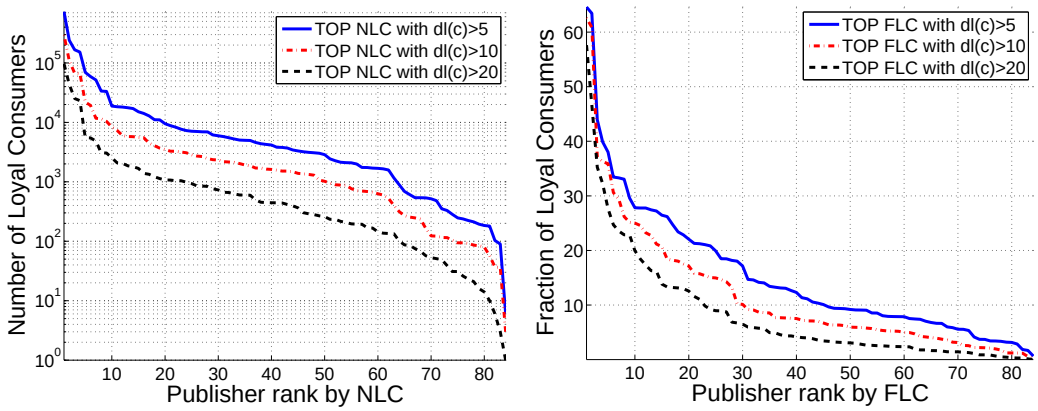
We define loyalty for individual consumers. We need to introduce two metrics to assess different aspects of loyalty of consumers towards a particular publisher p as follows:

- $NLC(p)$: The absolute number of loyal consumers for p,
- $FLC(p)$: The fraction of p's consumers that are loyal.

For publisher p, $FLC(p)$ indicates what fraction of p's consumers is loyal to p while $NLC(p)$ measures how many loyal consumers p has. To clarify the relation between these two metrics, let's consider the following simple example: publisher p_1 with 1000 consumers and $NLC(p_1)$ = 100 has $FLC(p_1)$ = 0.1 whereas publisher p_2 with 200 consumers and $NLC(p_2)$ = 100 has $FLC(p_2)$ = 0.5. Both p_1 and p_2 are able to attract the same number of loyal consumers, however the *strategy* used by p_2 seems to be more effective since a higher percentage of all its consumers is loyal.

Figure 5.11(a) and 5.11(b) depict the distribution of NLC and FLC across all publishers in each one of the target groups, respectively. Furthermore, each figure also plots the specified distribution by considering only a subset of consumers whose $dl(c)$ is larger than 5, 10 and 20.

These figures demonstrate that the top-100 publishers not only have a significantly larger number of loyal consumers but also have a larger fraction of loyal consumers. Note that as we focus on more active consumers (with larger $dl(c)$), the



(a) NLC (b) FLC

Figure 5.12: $NLC(p)$ and $FLC(p)$ for non-fake top-100 publishers. x axis indicates the publisher's rank based on the target metric among publishers.

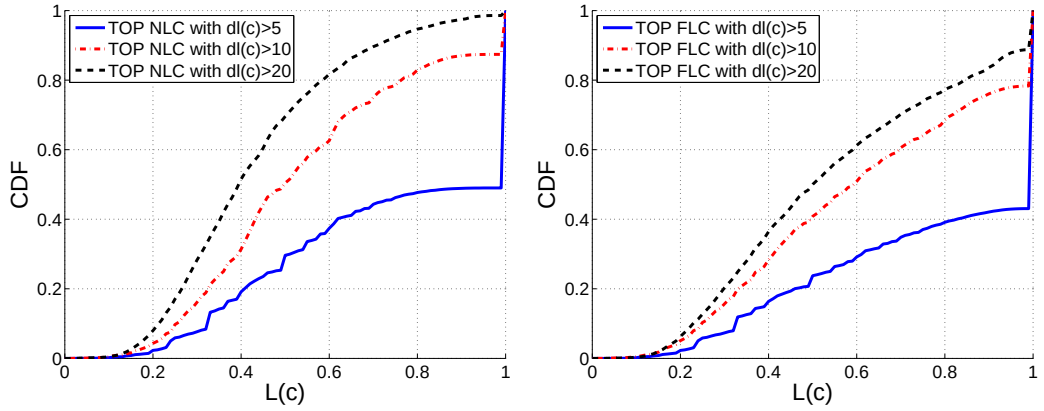
distribution of NLC and FLC maintain the same shape but shift towards lower values for both groups of publishers. This suggests that the observed trends among target group of publishers do not significantly change by considering consumers with different level of activities.

For the rest of this Section, we focus on the top-100 publishers since a majority of all loyal consumers are associated with these publishers. In particular, 72%, 64% and 48% of consumers are associated with these publishers for $dl(c)$ larger or equal to 5, 10 and 20, respectively. In our analysis, we will also leverage the business profile of these publishers that we determined in Section 5.5.

5.7.3 Loyalty Towards top-100 Publishers

Focusing our analysis on the top-100 publishers, Figure 5.12(a) and 5.12(b) show the value of $NLC(p)$ and $FLC(p)$ across these publishers, respectively. In these figures publishers across the x-axis are ranked by their $NLC(p)$ (or $FLC(p)$) value and each line shows the result using a different minimum $dl(c)$ value for filtering active consumers. Note that the y-axis in Figure 5.12(a) has log scale.

These figures reveal that the value of NLC and FLC across the top 100 publishers varies dramatically. In particular, the top-10 publishers with the largest NLC values collectively attract around 84% of all loyal consumers associated with all top-100 publishers (independent of the minimum level of activity among consumers). We call this group *top-NLC*. Furthermore, only top-10 publishers based on FLC have a significant fraction of loyal consumers (at least 14% to 28% for different $dl(c)$). We



(a) top-NLC (b) top-FLC

Figure 5.13: CDF of $L(c)$ across consumers of top-NLC and top-FLC publishers using different minimum $dl(c)$ to filter consumers

call this group *top-FLC*. Focusing on more active consumers obviously reduces the number of loyal consumers and thus the value of NLC and FLC for each publisher. However, increasing $dl(c)$ does not seem to generally change the overall trends of these results. This suggests that only top-NLC and top-FLC publishers appear to have a significant base of loyal consumers and thus we focus on these two groups[5].

Examinations of the identity of top-NLC and top-FLC publishers for different $dl(c)$ values revealed the following key points: First, there is only two overlapping publishers, namely *eztv* and *exmnova*, between two groups for $dl(c) \geq 5$. Second, as we focus on more active consumers ($dl(c)\geq$10, 20), we observe only two other overlapping publishers, namely *Rabiner* and *artpepper*, in top-NLC and top-FLC groups but they are ranked at the end of these groups.

In summary, our results show that consumer loyalty towards publishers (measured by NLC or FLC) is very skewed regardless of the minimum expected level of activity among consumers. Only a small number of top-NLC and top-FLC publishers appear to have a significant base of loyal consumers. However, most of the publishers in these two groups are unique. This suggests that top-NLC and top-FLC publishers are likely to exhibit different characteristics. Therefore, we investigate our two motivating questions for non-overlapping top-NLC and top-FLC publishers and their consumers.

[5]We have also identified and examined the top-10 publishers based on FLC across *all* publishers (not just top-100). While two of these publishers are in our top-FLC, the other eight publish a very small number of files and attract an insignificant number of consumers. Since their impact is negligible, we only focus on top-FLC.

dl(c)	Avg(L(c)) Top-NLC	Avg(L(c)) Top-FLC	Norm Diff
≥ 5	0.72	0.74	2.74%
≥ 10	0.54	0.61	12.17%
≥ 20	0.40	0.56	33.33%

Table 5.8: Average loyalty for consumers of top-FLC and top-NLC publishers

Fine-Grain Loyalty Towards top Publishers: So far we only considered $NLC(p)$ and $FLC(p)$ as two coarse measures of consumer loyalty towards publisher p. We now take a closer look at the level of loyalty by individual consumers (or $L(c)$) towards top-NLC and top-FLC publishers. Figure 5.13(a) and 5.13(b) depict the distribution of $L(c)$ among all loyal consumers of non-overlapping top-NLC and top-FLC publishers, respectively. Each figure shows the distribution for different minimum level of activity ($dl(c)$) among consumers as well. Comparing lines for similar $dl(c)$ values in these figures demonstrates that *the top-FLC publishers not only attract a higher fraction of loyal consumers but the level of loyalty among their consumers is relatively higher than top-NLC publishers*.

To better demonstrate this point, we use the average value of $L(c)$ across consumers of publishers in each group (Avg(L(c)). Since the value of $L(c)$ is always between 0 and 1, average $L(c)$ provides a useful indicator to compare two groups. Table 5.8 summarises the average value of $L(c)$ across consumers for top-NLC and top-FLC publishers using different $dl(c)$ values for filtering. The last column of Table 5.8 presents the normalised difference in average $L(c)$ between two groups. This table clearly shows the following points: *(i)* the average loyalty among consumers of top-FLC consumers is higher than consumers of top-NLC for any minimum level of activity among consumers. *(ii)* the value of $NormDiff$ reveals that the gap between loyalty of consumers grows as we focus on more active consumers.

USERNAME	BUSINESS	TYPE OF PUBLISHED CONTENT	REPUTATION AT PIRATEBAY	NUMBER OF PUBLISHED CONTENT	NUMBER OF CONSUMERS	NLC(p)	FLC(p)	AVG(L(c))
eztv	PP	VIDEO (Tv Shows)	VIP	313	1,151,633	730,558	63.44	0.7532
exmnova	PP	PORN (Movies)	Trusted	1,780	633,995	241,003	38.01	0.6677
TvTeam	PP	VIDEO (Tv Shows)	VIP	2,332	799,035	166,497	20.84	0.7664
Rabiner	PROMO	PORN (Movies)	Trusted	662	559,526	152,342	27.23	0.7187
raymondhome	PROMO	VIDEO (Movies)	VIP	125	494,343	69,451	14.05	0.7816
VTV	A	VIDEO (Tv Shows)	VIP	119	632,672	59,029	9.33	0.7144
extremezone	PP	VIDEO (Movies)	VIP	47	221,430	51,832	23.41	0.8435
Housezz	A	AUDIO, APPS, VIDEO	Deleted	225	332,959	33,609	10.09	0.7054
pizstol	PROMO	PORN (Movies)	Deleted	461	292,708	33,191	11.34	0.6669
1.No.1	PROMO	PORN (Movies)	VIP	223	200,857	18,812	9.37	0.5563

Table 5.9: Main characteristics of Top-NLC publishers for dl(c)$\geq$5; PP: Private (BitTorrent) Portal, Promo: Promoting Web Site, A: Altruistic

USERNAME	BUSINESS	TYPE OF PUBLISHED CONTENT	REPUTATION AT PIRATEBAY	NUMBER OF PUBLISHED CONTENT	NUMBER OF CONSUMERS	NLC(p)	FLC(p)	AVG(L(c))
ClaudiaZ	A	VIDEO (Tv Shows)	VIP	162	7,730	4,994	64.61	0.7882
eztv	PP	VIDEO (Tv Shows)	VIP	313	1,151,633	730,558	63.44	0.7532
Mois20	PP	VIDEO (Tv Shows)	VIP	250	38,919	17,099	43.94	0.6677
CanadaJoe	A	AUDIO	VIP	401	14,458	5,768	39.90	0.7853
exmnova	PP	PORN (Movies)	Trusted	1,780	633,995	241,003	38.01	0.6677
mikexxxryan	A	PORN (Movies)	Deleted	61	17,796	5,952	33.45	0.8645
starburst3	A	PORN (Pictures)	Deleted	133	7,315	2,436	33.30	0.6568
0oEdito0	A	COMICS	Deleted	73	18,616	6,153	33.05	0.5780
SkullManWoopt	A	PORN (Picture)	Trusted	69	12,502	3,702	29.61	0.7527
ESPALPSP	A	GAMES	Deleted	90	31,728	8,829	27.83	0.8081

Table 5.10: Main characteristics of Top-FLC publishers for dl(c)$\geq$5; PP: Private (BitTorrent) Portal, Promo: Promoting Web Site, A: Altruistic

Attribute	top-FLC	top-NLC
FLC(p)	27-65% (33%)	9-27% (13%)
NLC(p)	3k-17k (6k)	18k-166k (55k)
# Pub. Files	61-401 (111)	47-2332 (224)
# Consumers	7k-39k (16k)	200k-799k (413k)
Profile Type	Altruistic	Private Tracker
Content Type	Foreign, Comic	Movie/TV Show

Table 5.11: Main characteristics of top-FLC and top-NLC publishers

5.7.4 Top-NLC vs. Top-FLC Publishers

Our hypothesis is that top-FLC and top-NLC publishers exhibit different business profiles which in turn results in their high FLC or NLC values. To explore this hypothesis, we examined the following key attributes of these two groups of publishers: their username, business profile (as we determined in Section 5.5), type of posted content, assessed reputation by the Pirate Bay portal, number of published content, and total number of their consumers. In particular, the level of reputation for each publisher is assigned by the Pirate Bay based on the past history of the publisher on this portal. These levels from high to low are: VIP, Trusted, Helper and Unknown (i.e., no reputation is assigned). Since the information about individual publishers was collected a few months after our main data collection, some of the publishers have left the Pirate Bay. The reputation of these departed publishers was set to *Deleted*. Table 5.9 and 5.10 provide detailed characteristics of top-NLC and top-FLC publishers. For easier comparison, Table 5.11 summarises the range of these characteristics (with the median value in parentheses) for non-overlapping publishers in both groups.

Interestingly, uncommon publishers in each group exhibit rather distinct characteristics. Non-overlapping top-NLC publishers are mostly profit driven publishers that publish 47-2332 popular content (e.g., recent episodes of popular TV shows, recent Hollywood movies or porn videos). In addition, the ranges of FLC(p) and NLC(p) value for these publishers are 9-27% and 18-166K, respectively. In contrast, non-overlapping top-FLC publishers are mostly altruistic publishers who upload a small to moderate number (61-401) of rather specialised content (e.g., movies in a non-English language, comics or porn pictures). Their published content is not as popular as top-NLC publishers, thus they attract a smaller number of consumers and therefore a smaller number of loyal consumers (3-17K). However, a larger fraction

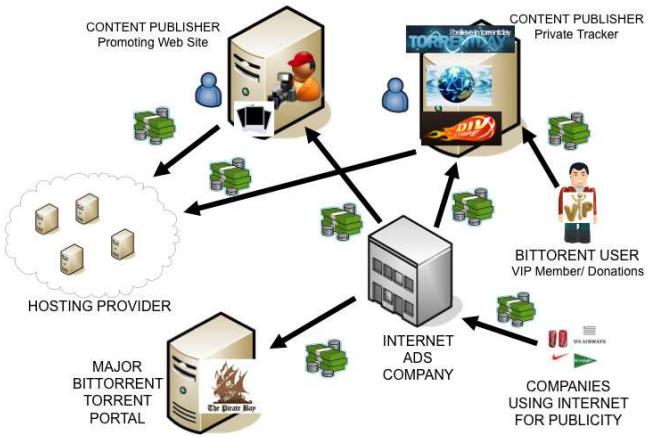


Figure 5.14: Business Model of Content Publishing in BitTorrent.

(FLC(p)) of their consumers are loyal and exhibit a rather larger level of loyalty than consumers of NLC(p). The overlapping publishers appear to exhibit a combination of these characteristics, which results in their appearance in both groups. *In summary, top-FLC and top-NLC publishers exhibit a different characteristics in terms of the number, type and popularity of published content that lead to a different loyalty pattern among their consumers.*

5.8 Other Beneficiaries in BitTorrent Marketplace

In previous Sections we analysed the main characteristics of major content publishers in BitTorrent, demonstrating that content publishing is a profitable *business* for an important fraction of the top publishers. While we have focused primarily on content publishers, there are other players around the BitTorrent ecosystem [118] that have financial interest and may promote this marketplace around BitTorrent. These other beneficiaries include: *Major BitTorrent Portals*, *Hosting Providers* and *Ad companies*. Figure 5.14 shows the interactions between different players in the BitTorrent marketplace where the arrows indicate the flow of money between them. In this Section, we briefly describe the role of main players and their interactions with others

Major Public BitTorrent Portals such as the Pirate Bay are dedicated to indexing torrent files. They basically serve as rendezvous points for content publishers and consumers. The main advantage of these major portals is the reliable access (e.g., by rapidly removing fake or infected content) to popular content. This motivates millions of BitTorrent users to visit these portals every day, which in turn makes these

web sites very valuable. For instance, the Pirate Bay is one of the most popular sites across Internet (ranked the 77th in the Alexa Ranking as of November 15th 2011) and is valued around $10M.

Hosting Providers are companies dedicated to renting servers. Heavy seeding activity performed by some publishers requires significant resources (e.g., bandwidth and storage). Thus, a large fraction of major publishers rent servers from hosting providers that generates income for hosting providers proportional to the level of activity by the publisher. For example our measurement revealed that around 78 to 164 servers (i.e., unique IP addresses) associated with major publishers are hosted at a single ISP in France, called OVH. Considering the cost of an average server at OVH (around 300 €/month from OVH web site), we estimate that its average monthly income from BitTorrent publishers is between 23K to 43K €/month. It is worth noting that some hosting providers (e.g., Server Intellect) have adopted strict policies against P2P applications using their servers to distribute copyrighted material due to possible legal implications [24]. However, our exchange with OVH revealed that they do not monitor the activities performed by their customers and may react only when a violation is reported by a third party and if the related activity is not ceased by the customer [20]. This reactive and rather soft policy appears to have attracted publishers of copyrighted content to OVH.

Ad Companies pair customers who wish to post ads on the Internet with popular web sites where ads can be placed. These companies dynamically determine to which, typically popular, web site and when each ad is placed. The ad company and the web site both receive portion of this income. By attracting users through major BitTorrent portals, content publishers can increase the number of visits to their web sites and thus become a more desirable target for posting ads. We have examined the header of exchanged http messages between the browser and the publishers' web sites and verified that these web sites indeed host ads. Unfortunately, we are unable to estimate the level of income that publishers have from hosting ads.

5.9 Conclusions

In this Chapter we presented a measurement study on the largest BitTorrent portal to investigate socio-economic incentives among content publishers. Our results revealed that a small fraction of publishers are responsible for two-thirds of the published content and three quarters of the downloads. Our careful investigation on the incentives of major publishers in the largest BitTorrent portal led to the follow-

ing important findings. First, anti-piracy agencies and malicious users perform a systematic poisoning index attack over major BitTorrent portals by publishing fake content in order to obstruct download of copyrighted content and to spread malware or scam the users, respectively. Roughly, one-third of the published content and a quarter of all downloads are associated with fake content. This finding indicates that BitTorrent portals can be leveraged by malicious users to easily perform their malicious activity to a large number of users, which could be a major security concern. Moreover, just a few tens of users are responsible for most of the published fake content. Furthermore, more than 99% of the fake torrents are associated with either malware or scam websites. This represents a serious threat for the BitTorrent ecosystem that must be eliminated or at least mitigated. We address this issue in next Chapter. Second, excluding the fake publishers, the remaining top publishers are responsible for one-third of all published content and half of all the downloads. Our evidences suggest that half of these top publishers leverage the published copyrighted content on BitTorrent portals to attract content consumers to their web sites for financial gain. We also demonstrate that these profit-driven publishers exhibit clearly distinct characteristics (i.e., a signature) compared to altruistic publishers. Third, we examined consumer loyalty toward top publishers and showed that the altruistic publishers attract a larger fraction of loyal consumers with a higher level of loyalty compare to profit-driven publishers. Overall, our study sheds an insightful light on socio-economic factors that seem to drive the popularity of BitTorrent and thus could affect the significant impact of its associated traffic on the Internet.

Chapter 6

Web Application for BitTorrent

6.1 Introduction

This Chapter describes the publicly accessible web portal [11] which we created as an outcome of this thesis. We have designed a system which continuously monitors the BitTorrent ecosystem and which provide interface to display gathered information through the web page. The portal provides information on two main areas of our research *(i)* BitTorrent content publishers and *(ii)* fake content in BitTorrent ecosystem. The first part, called MYPROBE, allows the user to learn about top publishers in BitTorrent. It is possible to check who published the most content in last 30 days and to get more details about the publisher like from which country he came or which ISP he uses. In the second part, we provide access to our system called TorrentGuard, which allows BitTorrent downloaders to check if the given torrent is linked with a fake content. Detailed information about each of those parts can be find in next Sections.

6.2 Software for Content Publishing Monitoring

In order to make our measurement techniques and our findings more accessible to other researchers and BitTorrent users, we have integrated our measurement tools into a system called *Monitoring, identifYing & PROfiling BitTorrent publishErs (MYPROBE)*. MYPROBE continuously monitors the publishing activity in the Pirate Bay portal by implementing the measurement methodology that we described in Section 5.2. In particular, it leverages the RSS feed to quickly detect a newly published content and then retrieves the following information for a detected torrent:

filename, content category and subcategory (based on the Pirate Bay categories), publisher's username, and (in those cases that we can) the publisher's IP address as well as the ISP, City and Country associated to this IP address. The main characteristics of any identified torrent and its publisher are stored in a database with a web-based portal [12] that allows users to query and obtain information. Our application has two goals. On the one hand, we want to share this data with the research community to permit further analysis of different aspects of the BitTorrent content publishing activity. On the other hand, we believe that this application can be useful for regular BitTorrent clients. First, a BitTorrent client can easily identify those publishers that publish content aligned with her interest (e.g., an e-books consumer could find publishers responsible for publishing large numbers of e-books). Furthermore, we have designed and implemented a mechanism to identify and filter fake publishers [26], allowing BitTorrent users of our application to avoid downloading fake content. This application would be described in next Section.

6.3 Software for Detecting Fake Content

In the previous Sections we have demonstrated that a large number of fake content (up to 35%) is currently being published in the BitTorrent ecosystem, and what is worse, most of these fake contents are potentially harmful for those users that download them. We have also seen that the techniques used to remove these contents are inefficient and requires human intervention to: first, detect and report the falseness of a given content, and second, to remove it from the BitTorrent portals (this is done by the portal administrator). Furthermore, the scope of the user reports is limited to a single BitTorrent portal, thus the content is removed exclusively from this portal instead of the whole BitTorrent ecosystem.

In this Section we present our tool, named TorrentGuard [76], which aims to automatise and accelerate the process of detecting fake publishers. For this purpose, TorrentGuard identifies a fake publisher by its IP address instead of its username as it is done by BitTorrent portals, such as the Pirate Bay, nowadays. By doing so, a fake content can be identified just after its birth since we can identify that the IP address of the initial seeder belongs to a fake publisher. This allows to accelerate the detection process.

Furthermore, contrary to current techniques used by BitTorrent portals, TorrentGuard removes the fake content from the whole BitTorrent ecosystem because it reports the content infohash. Since the infohash uniquely identifies a content in the

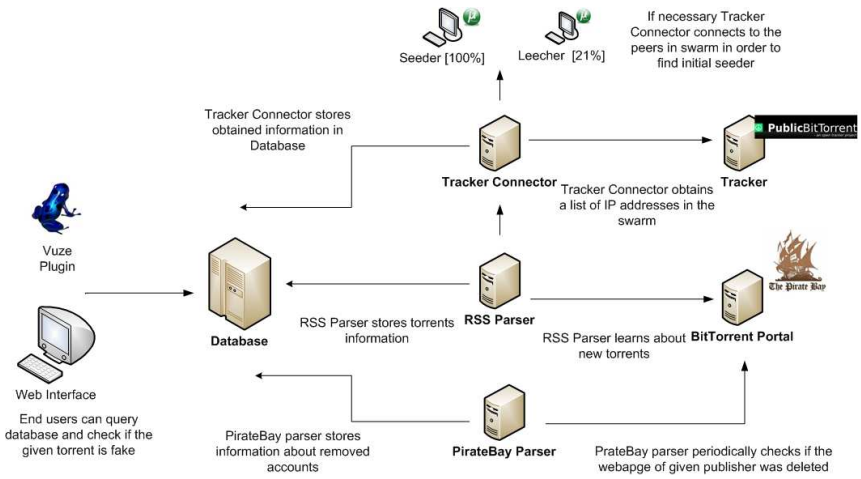


Figure 6.1: The schema of TorrentGuard

BitTorrent ecosystem, a user of TorrentGuard can identify the content as fake independently of the portal from which the content was retrieved (or even if it comes from the BitTorrent DHT service).

In the rest of the Section we present the details of the TorrentGuard implementation as well as the performance results obtained over a testing period of 14 days.

6.3.1 TorrentGuard Implementation

Figure 6.1 depicts a complete schema of TorrentGuard. It is composed by the following modules:

- *RSS Parser*: this module continuously monitors the RSS feed of The Pirate Bay portal. For each new published torrent the RSS Parser gathers the content infohash, from either the .torrent file or the magnet link[1], and also the publisher's username. Furthermore, the RSS Parser sends requests to the Tracker Connector.

- *Tracker Connector*: this module is responsible for connecting to the tracker for every torrent obtained by the RSS Parser. The main objective of the Tracker Connector is to obtain the IP address of the initial seeder. In those swarms where the list of IP addresses returned by the tracker contains more peers than just one seeder, this module connects to all the peers and retrieves their

[1]From 1st of March 2012, our tool uses exclusively magnet links for this purpose, as the Pirate Bay stopped serving .torrent files from that date.

bitfield in order to identify which one is the initial seeder. If the IP address of the initial seeder matches with one of those included in the blacklist of fake IP addresses, this torrent is marked as fake.

- *The Pirate Bay Parser*: this module periodically connects to the Pirate Bay web page associated to the different discovered publishers. Eventually, when a publisher's web page (i.e., account) is removed from the Pirate Bay, The Pirate Bay Parser marks this username as fake.

- *Database*: It stores all the relevant information for the detection and evaluation of TorrentGuard. For each inspected torrent it stores detailed information such as the publisher's username and the initial seeder IP address (in case this is possible to obtain). More importantly, it includes two blacklists. The first one contains the infohashes of all the discovered fake torrents whereas the second one includes the IP addresses of fake publishers found so far.

- *Website Interface* and *Vuze plugin*: The TorrentGuard functionality is publicly available throughout two different interfaces: a website [26] and a Vuze plugin. These interfaces provide access to the blacklist of fake torrents allowing a user to verify if a torrent file is associated to a fake content before starting the download process.

Next, we describe the functionality of TorrentGuard. It uses the Pirate Bay portal in order to identify new fake publishers and the IP addresses from where they operate. Towards this end, the RSS Parser continuously monitors the RSS feed of the Pirate Bay portal to learn about new torrents and identify for each torrent the publisher's username. Furthermore, it sends a query to the Tracker Connector that retrieves the IP address of the initial seeder (if it is possible). Both, the publisher's username and IP address (i.e., IP address of the initial seeder) are stored in database. In parallel, The Pirate Bay Parser periodically connects to the web page of the different discovered publishers within the Pirate Bay. If we find that a publisher's account is removed, this user and all its torrents are marked as fake. In addition, we annotate this publisher's IP address as *potential fake IP address*. If three different accounts associated to a given publisher's IP address are removed from the Pirate Bay, we consider that IP as a *fake IP address*. From this moment on, any content published from that IP address is identified just after its birth and reported as fake. Therefore, in the worst case, i.e., for new fake publishers, TorrentGuard employs the same time as the Pirate Bay to identify fake contents. However, once the fake publisher's IP address has been identified, TorrentGuard is able to report fake content immediately

after its publication what provides a significant improvement compared to standard detection mechanisms. Moreover, with TorrentGuard it is not necessary to manually report each fake content. Besides, three reports can be enough to mark the malicious user as a fake and in consequence all its future torrents will be automatically classified as fake.

Furthermore, the current existing solutions are limited to the portal where they operate. For instance, in the case of the Pirate Bay, once a content is identified as fake it is removed from the portal but not from the BitTorrent Ecosystem. Rather, TorrentGuard is a cross-portal solution, that is able to identify the infohash of the fake content preventing its download independently of the source from where the user obtained the .torrent file: any BitTorrent portal or the DHT service.

In short, TorrentGuard is a novel tool that: (i) reduces fake content detection time since it uses IP-based detection instead of username-based detection and (ii) allows to identify a fake content in the whole BitTorrent ecosystem rather than in a single portal because it identifies the fake content using the infohash (a unique identifier of the content in the whole BitTorrent ecosystem).

6.3.2 TorrentGuard Performance

We have evaluated the performance of TorrentGuard and compared it with the fake content detection mechanism used by the Pirate Bay during a testing period of 14 days. First, we count how many fake content published in the Pirate Bay are identified by the TorrentGuard just after its birth. Furthermore, we measure how long the Pirate Bay takes to identify these fake content. The obtained results show that TorrentGuard is able to early detect around 50% of the fake content uploaded to the Pirate Bay. Moreover, Figure 6.2 represents the CDF of the time difference between the detection instant of TorrentGuard and the Pirate Bay for these content. We observe, that TorrentGuard reduces the detection time 60 minutes in median. However, this reduction is higher than 2 hours for 20% of the fake contents, and for some cases it goes up to several days.

Although previous results already demonstrate the significant improvement provided by our tool compared to the state of the art solution, the final objective of TorrentGuard is reducing the number of download events associated with fake content, thus preventing BitTorrent users facing malware and scam. Then, if TorrentGuard was widely used, it would have prevented almost 390K fake content downloads just during the 14 days of the evaluation period compared to the Pirate Bay. By extending this value to a complete year, we can state that TorrentGuard would be able to elimi-

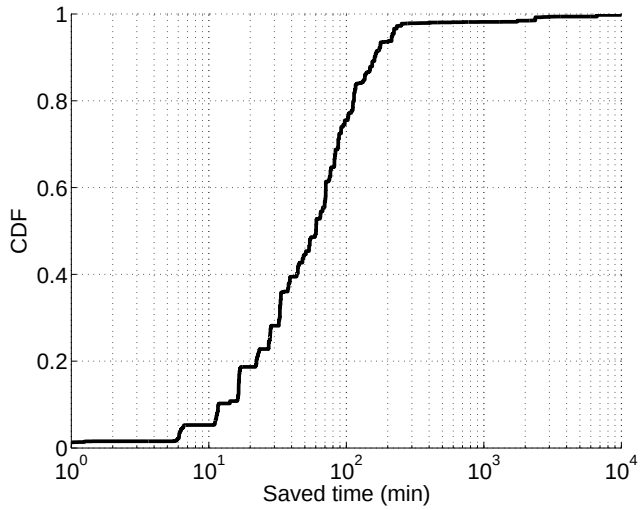


Figure 6.2: CDF of the saved time in fake content detection when using TorrentGuard in front of the Pirate Bay

nate more than 10 millions fake content downloads per year compared to the existing the Pirate Bay solution. However, as stated before the Pirate Bay solution is specific for this portal but it is not applicable to the whole BitTorrent ecosystem. Specifically, in our dataset we identify around 950K fake content downloads occurring after the Pirate Bay identifies these content as fake. Rather, our proposed solution would be able to avoid also these downloads. Overall, TorrentGuard could avoid more than 1.35 millions fake content downloads in a period of two weeks. This means more than 35 millions in the course of a year. Finally, it is worth to mention that even this impressive number is only a lower bound since in our evaluation we only consider download events associated to few of the most important BitTorrent Trackers[2] but we do not consider download events coming from minor BitTorrent Trackers or the BitTorrent-associated DHT systems.

In a nutshell, our initial evaluation suggests that TorrentGuard could avoid up to tens of millions fake downloads per year. More importantly, this supposes (depending on the success of the fake publishers' strategies) up to hundreds of thousands of computer infections and scam episodes. Hence, our evaluation shows very promising results to incentive the BitTorrent community to use the TorrentGuard.

[2]For instance, `http://openbittorrent.com/`, `http://publicbt.com/` that are the two major Trackers in the BitTorrent ecosystem

6.3.3 TorrentGuard Efficiency

A detection system is typically characterised by the number of false negative and false positive occurrences. On the one hand, the former is represented by those fake torrents escaping our detection tool. On the other hand, false positives refer to those content classified as fake, which actually are non-fake ones. Unfortunately, it is not feasible to properly measure such parameter since it would require to manually inspect a huge amount (thousands) of contents classified as legacy (i.e., non-fake) ones. Instead, we have performed an affordable evaluation by downloading few dozens of torrents classified as legacy by TorrentGuard and we did not find any fake torrent among them. We can state, however, that our tool discovers all fake contents which are also detected by the Pirate Bay.

In order to evaluate the false positives rate of TorrentGuard, we focus on those Pirate Bay usernames whose account has not been deleted from the Pirate Bay but their content have been classified by TorrentGuard as fake. The first intuition is that TorrentGuard may be mistaken for some of these usernames. We have downloaded content from each of these referred Pirate Bay accounts and we did not find any non-fake content among them, thus these content belong to fake publishers that have still not been detected by the Pirate Bay.

Hence, the performed evaluation suggests that TorrentGuard suffers from a negligible rate of both false positive and false negatives.

6.3.4 Low impact of TorrentGuard External Dependencies

In this Section we discuss the external dependencies of TorrentGuard and demonstrate that they represent a minor limitation for the system.

6.3.4.1 Dependency in The Pirate Bay

We have explained above that TorrentGuard bases its operation in the Pirate Bay portal. We selected the Pirate Bay because it is the most important portal and one of the key elements of the BitTorrent ecosystem [118]. Fake publishers could use other portals in order to not be detected by TorrentGuard, but then their visibility would be significantly affected. As future work, we plan to extend TorrentGuard to other portals. The requirements for these portals are: (i) having a service to announce new published torrents (e.g., RSS or a web page) and (ii) having a system to report fake publishers (e.g., removing their accounts as in the Pirate Bay or marking fake content with special flags). It is worth to mention, that these two requirements are

pretty standard and widely offered by the most significant BitTorrent portals such as Mininova [19] or IsoHunt [16].

6.3.4.2 Dependency on Users' Reports

To the best of the authors' knowledge none existing software has the capacity of identifying a fake content under this context, i.e., the software should discern if the content is fake or not using as input the title of the content. For this purpose, we require the intervention of a human being. Hence, in practice we need at least one user's report to identify a fake content and its associated fake publisher. As discussed earlier, TorrentGuard can be configured to mark a fake publisher's IP address after the first user report (i.e., removed fake username account). However as stated before we prefer be more conservative and mark the IP as fake after 3 reports to minimise the false negatives.

In summary, the previous discussion demonstrates that the external dependencies of TorrentGuard do not affect seriously its performance. On the one hand, the dependency of TorrentGuard in a single portal can be overcome by extending the operation of TorrentGuard to multiple portals. It is worth to mention that the effectiveness of TorrentGuard will be directly related to the significance of the associated portals. On the other hand, the dependency on users' reports is inherent to any fake content detection system and cannot be removed until new semantic-enhanced software is implemented. Hence, the best we can do is minimise the dependency in users' reports and TorrentGuard achieves this objective.

6.3.5 Limitations of Potential Countermeasures to TorrentGuard

If TorrentGuard becomes widely used, it is likely that the fake publishers will react by defining new strategies (i.e., *countermeasures*) that allow them to escape the control of TorrentGuard. Our tool identifies the fake publisher based on the IP address that it uses to publish the fake content. Hence, the fake publishers can use two reactive strategies. First, they can try to hide their IP address and second, they can try to perform their activity from a large number of IP addresses. In this subsection, we will discuss these strategies and their potential effectiveness.

6.3.5.1 Hiding the Fake Publisher's IP address

The most straightforward way to hide an IP address is the utilisation of a proxy. In this case TorrentGuard will interpret that the fake activity is being performed from the proxy IP address and will banned this one. Hence this technique is not efficient against TorrentGuard.

The next option would be to consider a network of proxies such that the fake Publisher can use different proxies for publishing different fake content. This type of anonymisation services exist in the current Internet and are commonly used by regular BitTorrent users to hide its IP address during the process of illegal content downloads and TOR is an example [25]. In TOR, traffic from a source (a fake publisher in our case) is bounced through several relays until it reaches the destination. Hence, for the destination that packets are coming from the IP address of the last (or *egress*) proxy and the IP address of the source cannot be identified. Furthermore, the egress proxy changes from one communication to another. Fake publishers could exploit the functionality of TOR to avoid its IP address being detected by TorrentGuard. TorrentGuard would then mark the IP addresses of TOR egress proxies as fake. Hence, if some non-fake publishers would use TOR, TorrentGuard would also mark their content as fake, thus increasing the false positives rate.

However, it is important to highlight that these anonymity services were not designed for supporting heavy traffic applications such as BitTorrent so that the performance offered to these services is typically poor. Indeed, TOR developers specifically state that TOR does not perform well with BitTorrent and is not designed for handling that type of traffic [8]. To evaluate the performance degradation that a fake publisher would experiment using TOR we have run a very simple test that compare the performance of a regular BitTorrent download vs. a download done with usage of TOR. For this purpose we have chosen a mid-popular torrent from the Pirate Bay (around 200 seeders and 300 lechers, 350,5 MB) and downloaded it 10 times with and without TOR usage. We have run the experiment in premises of our University (with a symmetric connection of 100 Mbps) and using a home ADSL (with a download and upload bandwidth of 6 Mbps and 320 kbps respectively). The results are presented in Tab. 6.1. They suggest that operating BitTorrent over TOR reduces the performance around 3 times independently of the speed of the access link. Therefore, the utilisation of anonymisation networks by fake publishers would severely impact the performance (i.e., content download time) of the swarms associated to fake content. This would result in attracting a lower number of victims that would prefer faster downloads. In addition, we have revealed in Section 5.6.2 that the Top

Type of connection	Average Time	Average speed
University	6m 46s	6.9 Mbit/s
Home ADSL	9m 59s	4.68 Mbit/s
University (with TOR)	20m 31s	2.27 Mbit/s
Home ADSL (with TOR)	31m 15s	1.49 Mbit/s

Table 6.1: Average speed and download time of the file using BitTorrent with and without TOR

Fake Publishers perform their activity from high-speed services. This suggests that performance is a key aspect for their activity, thus anonymisation services seem to be a not appropriate option for them.

In summary, current solutions that could be used by a fake publisher in order to hide its IP address are either not efficient (e.g., single proxy) or incur an important performance degradation that seems to not be adequate for the fake publishers' activity.

6.3.5.2 Using Multiple IP Addresses

The second countermeasure that a fake publisher could opt for is using a large number of IP addresses such that it always have undetected IP addresses to use for publishing fake content. Next, we estimate the number of IP addresses that a fake publisher would need to perform its activity in the presence of our tool. TorrentGuard identifies an IP address as fake after detecting 3 fake user accounts in the Pirate Bay. Thus, TorrentGuard marks a content as fake starting from the 4th account used by the publisher. We demonstrated in Section 5.6.2 that top 5% of fake publishers use in average 4 user accounts per day. Hence, a Top Fake Publisher would need roughly 1 IP address per day in order to perform its activity and avoiding being blocked by TorrentGuard. In addition, we have seen that the activity of these publisher is performed from high-speed servers located in data centres. Hence, these users would need to have access to around 30 IP addresses associated to high-speed access links per month.

In short, this strategy represents a double serious challenge: first, the fake publisher should be able to get continuously 30 new IP addresses per month and second, these IP addresses needs to be associated to high-speed access links. This is rather difficult for regular Internet users and companies.

We can conclude that the studied countermeasures against TorrentGuard are either inefficient or unrealistic. Hence, the wide usage of TorrentGuard may lead to discourage fake publishers to perform their activity.

6.3.6 Torrent Guard Future Deployment

In the previous Sections we have demonstrated the enormous potential of our TorrentGuard prototype. However, we believe that there is still a room for improvement if BitTorrent portals and trackers get involved in a next stage for the development of TorrentGuard. In this case, TorrentGuard could be extended to be a distributed platform in which trackers would identify the IP address of the initial seeder for every content and BitTorrent portals would identify the infohash of fake torrents. BitTorrent portals would provide the infohash of fake torrents to trackers so that these would be able to blacklist the IP address associated to fake publishers and eliminate their associated swarms. Furthermore, trackers would report back to portals the infohash of every new fake torrent published from a blacklisted IP address so that portals can immediately remove the associated .torrent file. The described system could store the information in a central server that interacts with both portals and trackers and maintain a central repository that can be accessed by users as well. Another option is running a complete distributed system in which trackers and portals exchange the information without the necessity of any central server. We believe that the involvement of major BitTorrent portals and trackers in this project would lead to reduce the presence of fake content to negligible levels[3].

Currently TorrentGuard is accessible through web portal [15] where user can upload torrent file (or provide the infohash) to check if it is associated with fake content or not. Moreover, we have implemented plugin to Vuze (the most popular BitTorrent client), which automatically checks each torrent added by the user.

6.4 Conclusions

In this Section we have presented our tool which allows to monitor publishing phenomena in BitTorrent. Our web portal allows to find the heaviest publishers in BitTorrent and check their characteristics like number of published files or ISP from which they perform their activity.

[3]The authors have started a process to contact different Trackers and Portals to sense their interest in participating in the deployment of the described project.

Furthermore, previous Chapter presented that fake content represents a serious threat for the BitTorrent ecosystem that must be eliminated or at least mitigated. Towards this end, we have also implemented TorrentGuard, a novel tool for early detection of fake content. Based on our initial evaluation the widely usage of this tool may prevent the download of millions of fake content every year, thus contributing to reduce the number of computer infections and scam episodes faced by BitTorrent users.

Chapter 7

Summary and Future Work

7.1 Summary

In this thesis we have leveraged extensive measurement techniques to address different issues of BitTorrent with special focus on socio-economic aspects. First, we have made a classification of main measurement techniques which were applied in the past to analyse BitTorrent. Based on those techniques, we designed measurement tools that monitor BitTorrent and obtain various types of the data. We use different obtained dataset in order to make a study about BitTorrent. We put a special attention on three main characteristics: *(i)* connectivity properties of BitTorrent swarms *(ii)* behaviour and incentives of content publishers and *(iii)* mitigating the issue of fake content in the BitTorrent ecosystem.

First, we performed a detailed analysis about the topology structure and connectivity properties of live BitTorrent swarms. Our results demonstrate, that real BitTorrent swarms present a relatively efficient topology for the dissemination of information and are resilient to churn (i.e., random node removal process). However, real swarms are significantly less resilient to possible attacks (i.e., highest-degree node removal process) than equivalent random graphs. This is an important observation for those companies that use BitTorrent for critical/important functions (e.g., big software releases or content replications), since they can evaluate whether the resilience of BitTorrent to different events fulfil their security requirements. We also demonstrated that a significant fraction of peers presents more local neighbourhoods than expected from purely random neighbour selection process. This locality effect is marked even stronger at the exchange traffic level. Our results reveal that Indian ISPs along with large American and European ISPs are those hosting a larger number of users presenting a higher locality-biased in their neighbourhood compo-

sition. This suggests that locality-enforcing policies of some ISPs, the proliferation of locality-aware BitTorrent clients and some other network effects, such as congestion, are localising an important part of BitTorrent traffic within some ISPs. Those are important results for ISPs because they are usually interested in localising BitTorrent traffic in order to reduce the traffic in transit links what leads to reducing their operational cost. Finally, we performed an analysis of the composition of the neighbourhood of the peers and the results reveal that current BitTorrent implementations make both, leechers and seeders, modifying a significant portion of their neighbourhoods in short periods of time. This leads to a communication overhead that might not be needed. This might be especially important to consider in those cases where BitTorrent is used in low capacity devices, e.g., smartphones.

In the second part of the thesis we studied the content publishing activity in BitTorrent. We revealed that a small fraction of publishers are responsible for majority (67%) of the published content and, more importantly, for 75% of the downloads. First discovered group of the publishers publish legitimate files and have strong financial incentives for posting this content on BitTorrent portals. This presence of profit-driven publishers suggests that BitTorrent ecosystem relies on a handful of publishers. Because this group is responsible for more than a half published (non-fake) files, the removal of these financial-driven publishers (e.g., by anti-piracy actions) may significantly affect the popularity of these portals as well as the whole BitTorrent ecosystem. It may result in BitTorrent suffering from a high reduction in popularity. Second discovered group focuses on publishing fake files. On the one hand, there are malicious users which leverage BitTorrent to spread malware and this malware reaches millions of users. On the other hand, there are publishers who try to attract BitTorrent users to scam websites in order to get economical benefit from the victims by using different scam techniques. Finally, anti-piracy agencies publish fake content to protect copyrighted files. All of these fake publishers combined contribute 30% up to 35% of the content and attract 25% (several millions) of downloads. Moreover, more than 99% of the analysed fake files are linked to either malware or scam websites. This suggests that publishing fake content is a serious issue which should be eliminated or at least mitigated.

Finally, we have implemented a tool which continuously monitors BitTorrent ecosystem and gives access to obtained data through created web portal. The tool (called MYPROBE) presents a detailed information about top publishers from the Pirate Bay portal and about their activity. More importantly, as a part of this tool, we have implemented a software named TorrentGuard, which allows for early detection of fake content in BitTorrent. TorrentGuard may be accessed through web portal

or as a plugin to Vuze, the popular BitTorrent client. Our evaluation shows, that if extensively used, TorrentGuard can prevent the download of millions of fake content linked to malware or scam thus improve user experience.

7.2 Future Work

For future research we plan to extend our work keeping our focus on socio-economic aspects of BitTorrent. In particular, we plan to explore the three following areas: *(i)* effect of anti-piracy actions on the popularity of P2P *(ii)* recommendation system and *(iii)* popularity prediction.

First, the popularity of major P2P applications (including BitTorrent) is primarily due to the availability of copyright-infringing content. This, in turn, results in legal actions by copyright holders against P2P applications as well as new legislations and enforcement of anti-piracy laws. We want to characterise the trends in the population of publishers and consumers along with their level of activity in the BitTorrent ecosystem and investigate the effect of anti-piracy laws and their enforcement on the observed trends.

Secondly, recommendation system for BitTorrent would help to recommend which files are of interest to the users thus improving their experience. Our crawler gives us information about which user (identified by its IP) downloads which content. Based on it, we plan to define an algorithm that will be able to retrieve groups of BitTorrent users that are interested in the same content. The designed recommendation tool will be integrated together with TorrentGuard.

Finally, we want to focus our research on evolution of popularity of BitTorrent swarms. We want to analyse how the popularity of different torrents evolves over time. This could allow us to find different patterns of how the population of torrents changes during the time and, by linking this observations with other available information (like title or category of the torrent), to create a different groups of torrents. As a result, we want to build a tool that would analyse the first moments of torrent life and could classify it into one of those groups, thus being able to predict how the popularity of the torrent will change in the next part of torrent life.

7.3 Contribution

The contribution of this thesis in the area of BitTorrent measurements is a survey which present different measurement techniques:

- M. Kryczka, R. Cuevas, A. Cuevas, C. Guerrero, A. Azcorra: "Measuring BitTorrent Ecosystem: Techniques, Tips and Tricks", IEEE Communications Magazine, Vol. 49, Issue 9, pp. 144-152, September 2011

The main contributions in the area of BitTorrent connectivity, locality and resilience are:

- M. Kryczka, R. Cuevas, C. Guerrero, A. Azcorra: "Unrevealing the structure of live BitTorrent Swarms: methodology and analysis", IEEE International Conference on Peer-to-Peer Computing P2P 2011, Kyoto, Japan, 2011

- M. Kryczka, R. Cuevas, A. Cuevas, C. Guerrero, A. Azcorra: "Understanding the connectivity properties of real BitTorrent swarms and their implications in swarming efficiency, resilience and locality", under submission.

The contributions in the area of BitTorrent content publishers are following papers. First two of them focus on content publishers in general, while the third one describes fake content in BitTorrent:

- R. Cuevas, M. Kryczka, A. Cuevas, S. Kaune, C. Guerrero, R. Rejaie: "Is Content Publishing in BitTorrent Altruistic or Profit Driven?", The 6th International Conference on emerging Networking EXperiments and Technologies (CoNEXT), Philadelphia, USA, 2010

  This article had more than 170 references in media being referred in major Spanish TV, radio and newspapers.

- R. Cuevas, M. Kryczka, A. Cuevas, S. Kaune, C. Guerrero, R. Rejaie: "Unveiling the incentives for content publishing in popular BitTorrent portals", accepted to IEEE Transactions on Networking, this article was chosen to fast track from CONEXT conference.

- M. Kryczka, R. Cuevas, R. Gonzalez, A. Cuevas, A. Azcorra: "TorrentGuard: stopping scam and malware distribution in the BitTorrent ecosystem", under submission.

Finally we have created a portal (described in Section 6) which can be accessed at `http://bittorrent.netcom.uc3m.es` with MYPROBE and TorrentGuard modules (accessible through web portal and as a plugin to BitTorrent client Vuze).

Appendix A

The Representativeness of the Results

The BitTorrent ecosystem is large, complex and dynamic. Therefore, we cannot claim that the obtained results are valid for every portal in the BitTorrent ecosystem. However we state this is true for major BitTorrent portal.

Our study focused on the largest and thus most important BitTorrent portal in the BitTorrent ecosystem, namely the Pirate Bay. We have examined this claim and its implications to ensure that they are correct as follows.

First, we have identified the top 10 BitTorrent portals with the highest Alexa Ranking and obtained the maximum number of their daily visits as reported by Alexa. This information revealed that the Pirate Bay is the most visited portal and it receives at least twice more visits than the second largest portal, called Torrentz. Furthermore, Torrentz is a meta-search site that forwards the user to the portal that is actually storing the torrent such as the Pirate Bay. Therefore, if we consider the number of daily visits as an indicator for the size (and thus importance) of a portal, the Pirate Bay is by far the most popular portal in the BitTorrent ecosystem and properly represents large BitTorrent portals.

Second, a study by Zhang et al. [117] analysed private BitTorrent portals, called *darknets*, which are typically small. They estimate that the aggregate number of registered users across hundreds of the most important *darknets* is around 20M. Our *pb10* dataset alone contains a comparable number of users (around 27M IP addresses). This is another evidence that demonstrates the importance of the Pirate Bay portal in the BitTorrent ecosystem.

Third, another study by Zhang et al. [118] showed that major BitTorrent portals have a large fraction of overlap in their indexed torrents. For instance, they show that 71% of Mininova's and 75% of BTmonster's torrents (with more than 100 peers) were also indexed by the Pirate Bay at the time of their study. This observation

suggests that our findings are, at least, relevant to other major portals.

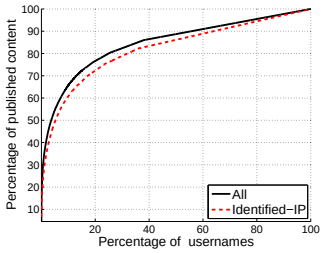
Appendix B

Potential Biases of the Measurement Methodologies

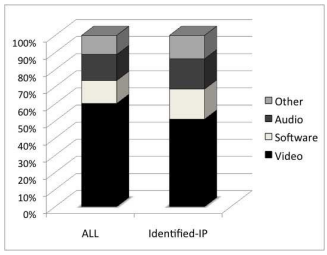
We can think of three possible sources of bias in our measurement: *(i)* excluding torrents whose publisher's IP address cannot be identified, *(ii)* capturing all downloaders in large torrents, *(iii)* impact of NAT on the identified consumers/downloaders. Next, we discuss these issues and show that the associated bias is insignificant and more importantly it does not affect the presented findings in the thesis.

1) Excluding Torrent Whose Publisher's IP Address Cannot Be Identified

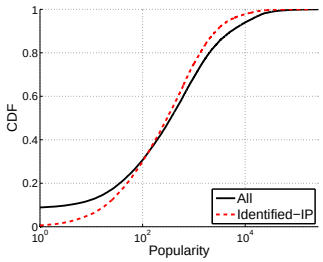
One potential bias in the analyses is the fact of using publishers' IP address since the corresponding dataset only includes 40% of torrents. As described in the first paragraph of Section 5.3, a publisher can be identified by either its username or its IP address (of the initial seeder). Username is a better identifier since it remains consistent across different torrents while IP address may change (e.g., a publisher using multiple servers to upload content). Therefore, we use the username as a user identifier in all of our analyses related to publishers except in Section 5.3.B and Section 5.4 where we must use the IP address of publishers. Since we are able to reliably determine the IP address of the initial seeder only for 40% of all the captured torrents, this could potentially introduce a bias in our results. To assess the level of this bias, we compare the following characteristics of this 40% of torrents (labelled as *Identified-IP*) with all torrents in the *pb10* dataset (labelled as All): (i) the distribution of the number of contributed files by the top x publishers (Figure B.1(a)), (ii) the breakdown of content type across contributed files in Figure B.1(b), and (iii) the distribution of torrent popularity in Figure B.1(c).



(a) *Percentage of content published by the top x% publishers for: All and Identified-IP publishers.*



(b) *Breakdown of content type across contributed files for: All and Identified-IP publishers.*



(c) *Distribution of torrent popularity for: All and Identified-IP publishers.*

Figure B.1: Comparison of main characteristics of *All* and *Identified-IP* torrents

These figures collectively show that these characteristics exhibit a very similar pattern with a very small difference for the two analysed groups. For example, the difference in the median value of both distributions in Figure B.1(a) is around 1.5%. The fraction of published video content is 10% more in *All* than in *Identified-IP*. Finally, the median popularity of torrents in Figure B.1(c) is 356 and 302, which represents a difference of 15%. It is important to note that our presented findings from the corresponding results in the thesis rely on the median value and overall shape of these distributions rather than their specific details.

In summary, our dataset for IP-based publishers analysis does not seem to exhibit a significant bias and the observed variations have minor effect on the presented figures and the associated findings.

2) Capturing All Downloaders In Large Torrents

Another possible source of bias is related to the ability of our measurement technique to capture all downloaders in a torrent. The typical allowed reconnection time by trackers is 10-15 minutes. Therefore, we used 8 distributed monitors where each one contacts the tracker for a target torrent (at least) every 15 minutes. In each contact, a monitor obtains the IP addresses of r random downloaders in the torrent where r = $min\{N, 200\}$ and N denotes the total number of peers within the torrent. To estimate the fraction of peers in a torrent that is captured by our approach, we conduct simulations considering a torrent with N peers where each query obtains r = $min\{N, 200\}$ random peers. Note that for each torrent in our dataset we consider N equals to the maximum reported size of the torrent what implies an unfavourable scenario. Furthermore, for our simulations we assume a simple churn model where each peer stays in the torrent for τ minutes (session time). We use our simulations

τ / ϕ	90%	95%	99%	100%
15 min.	92.7%	91.4%	89.1%	87.6%
1 h	96.0%	95.6%	94.9%	93.6%
2 h	98.8%	97.2%	96.0%	95.2%

Table B.1: Percentage of torrents for which our tool would be able to capture ϕ percent of the nodes considering different session times (τ) within our *pb10* dataset.

to determine the percentage of all users in a torrent that our approach is able to catch (ϕ) as function of the session time (τ) given the rate of our queries (32 query/hour). Our results in Table B.1 show the percentage of torrents within our *pb10* dataset for which our crawler would be able to crawl ϕ = 90%, 95%, 99% and 100% nodes considering different values of session time. These results suggest that even with a very short session time of 15 min., our tool would be able to collect all the nodes for more than 87% of the torrents in our dataset. If we consider a more realistic session time of 1 hour[1] this percentage increases to almost 94%. Again, we emphasise that the presented analyses that are affected by this issue (popularity of content in Section 5.4.B and Fig 5.2), only consider the average popularity across all torrents for each publisher. Therefore, the extremely popular torrents for which our tool may not be able to collect all the downloaders, have a very little impact in the presented results.

3) Impact of NATs in Downloaders Collection

Another factor that could affect our results are those users that are located behind NAT boxes. The presence of multiple users behind a NAT box would be revealed in our dataset by multiple peers with the same IP address but different ports. We examined the number of ports associated with each IP address in *pb10* dataset. We observed that more than 80% of the IP addresses appear with a single port whereas 95% of the IP addresses appear with one or two ports. We are unable to accurately determine the situation with those IP addresses that appear with multiple port numbers because they may indicate any of the following scenarios: (i) two (or more) users behind a NAT downloading the same content, (ii) two separate sessions associated with the same downloader. In case (ii), considering both IP addresses (with different ports) as a single user does not lead to error. Overall, these statistics shows the effect of NAT box on the estimation of consumers is not significant.

[1] 1 hour is the time needed to download a 450 MB file with an average download rate of 1 Mbps

Appendix C

Estimation of Session Duration

In this Appendix we explain the procedure utilised to calculate the duration of the session time of a given peer in a given torrent. We explain the procedure using the *mn08* dataset. Note, that it would be similar for other datasets.

Our *mn08* crawler connects to the tracker periodically and obtains a random subset of all the IP addresses participating in the torrent. Then, we cannot guarantee to obtain the IP address of the target peer in a resolution of seconds or even few minutes. This imposes some restrictions to compute the content publisher's seeding time in a given torrent.

Therefore, we firstly define a model to estimate the number of queries to the tracker (m) needed to obtain the IP address of the content publisher with a given probability $\mathcal{P}$. Let's assume that: (i) we have a torrent with $\mathcal{N}$ peers and (ii) for each query the tracker gives us a random set of $\mathcal{W}$ IP addresses. Then, if the target peer is in the torrent, the probability ($\mathcal{P}$) of obtaining its IP address in m consecutive queries to the tracker is given by:

$$\mathcal{P} = 1 - \left(1 - \frac{\mathcal{W}}{\mathcal{N}}\right)^m \tag{C.1}$$

We have computed the maximum instantaneous population of the torrents in our *mn08* and found that 90% of torrents have typically less than 165 concurrent peers. Then, we assume that the torrents have always a population $\mathcal{N}$ = 165. This is an upper bound that allows us to remove the noise introduced by the churn. We make a second conservative assumption: the tracker gives us $\mathcal{W}$ = 50 random IPs in each response (although in most of the cases we obtain 200 IP addresses). With these numbers and the proposed model we can assure that, if a peer (e.g., a content publisher) is in the torrent, we will discover it in m = 13 queries to the tracker with a probability higher than 0.99.

Next, we have calculated the time between 2 consecutive queries to the tracker in our dataset, and have checked that 90% of them are less than 18 minutes apart. Then, we again make a conservative assumption and consider that the time between two consecutive queries is 18 minutes.

Hence, multiplying the number of needed queries by the time between two consecutive queries we conclude that if a peer (e.g., content publisher) is in the torrent, we are able to get its IP address in a period of 4 hours with a probability equal to 0.99. Therefore, we consider that a given content publisher is off-line (i.e., its session has finished) if its IP address is not gathered in the torrent during 4 hours. We have repeated the experiments with 2h and 6h thresholds obtaining similar results.

Bibliography

[1] http://www.bittorrent.com/intl/es/company/about/ces_2012_150m_users.

[2] http://torrentfreak.com/facebook-uses-bittorrent/-and-they-love-it-100625/.

[3] http://torrentfreak.com/twitter-uses-bittorrent-for/-server-deployment-100210/.

[4] http://en.wikipedia.org/wiki/Magnet_URI_scheme.

[5] http://thepiratebay.se/blog/206/.

[6] http://www.theregister.co.uk/2008/06/02/onk_further_arrests/.

[7] http://yro.slashdot.org/yro/05/01/13/2248252.shtml?id=123&tid=97&tid=95&tid=1.

[8] https://blog.torproject.org/blog/bittorrent-over-tor-isnt-good-idea.

[9] Alexa rank. http://www.alexa.com/topsites.

[10] Amazon S3. http://aws.amazon.com/s3/.

[11] BitTorrent project portal. http://bittorrent.netcom.it.uc3m.es/.

[12] BitTorrent publishers portal. http://btpublishers.netcom.it.uc3m.es/.

[13] Captcha. http://www.captcha.org.

[14] Comparison of BitTorrent Clients. http://en.wikipedia.org/wiki/Comparison_of_BitTorrent_clients.

[15] Fakedetector. http://sourceforge.net/projects/fakedetector/.

[16] Isohunt. http://www.isohunt.com.

[17] Linux Tracker. http://linuxtracker.org/.

[18] MaxMind- GeoIP. http://www.maxmind.com/app/ip-location.

[19] Mininova. http://mininova.org.

[20] OVH's terms of service. http://www.ovh.co.uk/support/termsofservice/.

[21] PirateBay. http://thepiratebay.org.

[22] PirateBay RSS. http://thepiratebay.org/rss.

[23] Sandvine. Spring 2012 Global Internet Phenomena Report. http://www.sandvine.com/news/global_broadband_trends.asp.

[24] Server Intellect Use Policy. http://www.serverintellect.com/terms/aup.aspx.

[25] TOR project. http://www.torproject.org.

[26] TorrentGuard. http://torrentguard.netcom.it.uc3m.es/.

[27] Tribler. http://www.tribler.org.

[28] Bay TSP Annual Report, 2008. http://tech.mit.edu/V129/N28/piracy/BayTSP2008report.pdf.

[29] V. Aggarwal, A. Feldmann, and C. Scheideler. Can ISPs and P2P users cooperate for improved performance? *ACM SIGCOMM Comput. Commun. Rev., 37(3):29–40*, 2007.

[30] A. Al-Hamra, A. Legout, and C. Barakat. Understanding the properties of the BitTorrent overlay. Technical report, INRIA, Sophia Antipolis, July 2007.

[31] N. Andrade, M. Mowbray, A. Lima, G. Wagner, and M. Ripeanu. Influences on cooperation in BitTorrent communities. In *Proceedings of ACM SIGCOMM Workshop P2PEcon*, Philadelphia, USA, August 2005.

[32] A. Barabasi and E. Bonabeau. Error and attack tolerance of complex networks. *Scientific American Magazine, Vol. 288*, 2003.

[33] A. Bellissimo, B. N. Levine, and P. Shenoy. Exploring the use of BitTorrent as the basis for a large trace repository. Available at: `http://lass.cs.umass.edu/papers/pdf/TR04-41.pdf`, Technical Report 04-41, University of Massachusetts Amherst., 2004.

[34] A. R. Bharambe, C. Herley, and V. N. Padmanabhan. Some observations on BitTorrent performance. In *Proceedings of ACM SIGMETRICS '05*, Banff, Canada, 2005.

[35] A. R. Bharambe, C. Herley, and V. N. Padmanabhan. Analyzing and improving a BitTorrent network's performance mechanisms. In *Proceedings of IEEE INFOCOM '06*, Barcelona, Spain, 2006.

[36] R. Bindal, P. Cao, W. Chan, J. Medved, G. Suwala, T. Bates, and A. Zhang. Improving traffic locality in BitTorrent via biased neighbor selection. In *Proceedings of ICDCS'06*, Lisboa, Portugal, 2006.

[37] S. Le Blond, A. Legout, and W. Dabbous. Pushing BitTorrent locality to the limit. *Computer Networks Vol.55*, 2011.

[38] S. Le Blond, A. Legout, F. Lefessant, W. Dabbous, and M. Ali Kaafar. Spying the world from your laptop. In *Proceedings of LEET'10*, San Jose, USA, 2010.

[39] H. Chang, S. Jamin, Z. M. Mao, and W. Willinger. An empirical approach to modeling inter-AS traffic matrices. In *Proceedings of ACM IMC'05*, Berkeley, USA, 2005.

[40] Y. Ryn Choe, D. L. Schuff, J. M. Dyaberi, and V. S. Pai. Improving VoD server efficiency with BitTorrent. In *Proceedings of ACM MULTIMEDIA'07*, Augsburg, Germany, 2007.

[41] D. R. Choffnes and F. E. Bustamante. Taming the torrent: a practical approach to reducing cross-ISP traffic in peer-to-peer systems. In *Proceedings of ACM SIGCOM'08*, Seattle, USA, 2008.

[42] D. R. Choffnes, J. Duch, D. Malmgren, R. Guimera, F. E. Bustamante, and L. Amaral. Strange bedfellows: Communities in BitTorrent. In *Proceedings of IPTPS'10*, San Jose, USA, 2010.

[43] A. L. H. Chow, L. Golubchik, and V. Misra. Improving BitTorrent: a simple approach. In *Proceedings of IPTPS'08*, Tampa Bay, USA, 2008.

[44] B. Cohen. The BitTorrent protocol specification. *Available at:* `http://bittorrent.org/beps/bep_0003.html`, 2008.

[45] B. Cohen. Incentives build robustness in BitTorrent. In *Proceedings of First Workshop on Economics of Peer-to-Peer System*, Berkeley, USA, June 2003.

[46] F. Comellas, J. Ozon, and J. Peters. Deterministic small-world communication networks Information. *Information Processing Letters, Vol.76 No.1-2*, 2000.

[47] S. A. Crosby and D. S. Wallach. An analysis of BitTorrent's two Kademlia-based DHTs. Technical report, TR-07-04, Department of Computer Science, Rice University, June 2007.

[48] R. Cuevas, M. Kryczka, A. Cuevas, S. Kaune, C. Guerrero, and R. Rejaie. Is content publishing in BitTorrent altruistic or profit-driven? In *Proceedings of ACM CoNEXT'10*, Philadelphia, USA, 2010.

[49] R. Cuevas, M. Kryczka, A. Cuevas, S. Kaune, C. Guerrero, and R. Rejaie. Unveiling the incentives for content publishing in popular BitTorrent portals. *IEEE Transaction on Networking*, accepted to publication, 2012.

[50] R. Cuevas, N. Laoutaris, X. Yang, G. Siganos, and P. Rodriguez. Deep diving into BitTorrent locality. In *Proceedings of IEEE INFOCOM 2011*, Shanghai, China, 2011.

[51] C. Dale, J. Liu, J. Peters, and B. Li. Evolution and enhancement of BitTorrent network topologies. In *Proceedings of IEEE IWQoS'08*, Enschede, the Netherlands, June 2008.

[52] G. Dan and N. Carlsson. Dynamic swarm management for improved BitTorrent performance. In *Proceedings of IPTPS'09*, Boston, USA, 2009.

[53] C. Dana, D. Li, D. Harrison, and C.-N. Chuah. BASS: BitTorrent assisted streaming system for Video-on-Demand. In *Proceedings of IEEE 7th Workshop on Multimedia Signal Processing*, Shanghai, China, 2005.

[54] P. Dhungel, X. Hei, D. Wu, and K.W. Ross. A measurement study of attacks on BitTorrent seeds. In *Proceedings of ICC'11*, Kyoto, Japan, June 2011.

[55] P. Dhungel, D. Wuand B. Schonhorst, and K. W. Ross. Measurement and mitigation of BitTorrent leecher attacks. In *Proceedings of IPTPS'08*, Tampa Bay, USA, 2008.

[56] P. Dhungel, D. Wu, and K. W. Ross. Measurement and mitigation of BitTorrent leecher attacks. *Elsevier ComCom, Volume 32*, November 2009.

[57] M. Dischinger, A. Mislove, A. Haeberlen, and K P. Gummadi. Detecting BitTorrent blocking. In *Proceedings of ACM IMC'08*, Vouliagmeni, Greece, 2008.

[58] M. Dischinger, A. Mislove, A. Haeberlen, and K. P. Gummadi. Detecting BitTorrent blocking. *Proceedings of ACM IMC'08*, 2008.

[59] P. Erdos and A. Renyi. On random graphs. *Publicationes Mathematicae*, 1959.

[60] B. Fan, D.-M. Chiu, and J. C. S. Lui. The delicate tradeoffs in BitTorrent-like file sharing protocol design. In *Proceedings of ICNP'06*, Santa Barbara, USA, 2006.

[61] M. D. Fauzie, A. H. Thamrin, R. V. Meter, and J. Murai. A temporal view of the topology of dynamic BitTorrent swarms. In *Proceedings of IEEE NetSciCom'11*, Shanghai, China, 2011.

[62] L. Guo, S. Chen, Z. Xiao, E. Tan, X. Ding, and X. Zhang. Measurements, analysis, and modeling of BitTorrent-like systems. In *Proceedings of ACM IMC'05*, Berkeley, USA, 2005.

[63] D. Hales, R. Rahman, B. Zhang, M. Meulpolder, and J. Pouwelse. BitTorrent or BitCrunch: Evidence of a credit squeeze in BitTorrent? In *Proceedings of WETICE '09*, Groningen, the Netherlands, 2009.

[64] T. Hoifeld, F. Lehrieder, D. Hock, S. Oechsner, Z. Despotovic, W. Kellerer, and M. Michel. Characterization of BitTorrent swarms and their distribution in the internet. *Elsevier Computer Networks*, 55(5), 2011.

[65] C. Huang, J. Li, and K. W. Ross. Can internet video-on-demand be profitable. In *Proceedings of ACM SIGCOM'07*, Kyoto, Japan, 2007.

[66] T. Isdal, M. Piatek, A Krishnamurthy, and T. Anderson. Leveraging BitTorrent for end host measurements. In *Proceedings of PAM'07*, Louvain-laneuve, Belgium, 2007.

[67] M. Izal, G. Urvoy-Keller, E.W. Biersack, P.A. Felber, A. Al Hamra, and L. Garces-Erice. Dissecting BitTorrent: Five months in a torrent's lifetime. In *Proceedings of PAM '04*, Antibes Juan-les-Pins, France, 2004.

[68] R. Izhak-Ratzin, H. Park, and M. van der Schaar. Reinforcement learning in BitTorrent systems. In *Proceedings of IEEE INFOCOM'11*, Shanghai, China, 2011.

[69] S. Jun and M. Ahamad. Incentives in BitTorrent induce free riding. In *Proceedings of ACM SIGCOMM Workshop P2PEcon*, Philadelphia, USA, August 2005.

[70] A. Kalafut, A. Acharya, and M. Gupta. A study of malware in peer-to-peer networks. In *Proceedings of ACM IMC'06*, Rio de Janeiro, Brazil, 2006.

[71] T. Karagiannis, A. Broido, N. Brownlee, K. C. Claffy, and M. Faloutsos. Is P2P dying or just hiding? In *Proceedings of IEEE Globecom'04*, Dallas, USA, 2004.

[72] T. Karagiannis, P. Rodriguez, and K. Papagiannaki. Should internet service providers fear peer-assisted content distribution? In *Proceedings of ACM IMC'05*, Berkeley, USA, 2005.

[73] S. Kaune, R. Cuevas, G. Tyson, A. Mauthe, C. Guerrero, and R. Steinmetz. Unraveling BitTorrent's File Unavailability: Measurements, Analysis and Solution Exploration. In *Proceedings of IEEE P2P'10*, Delft, the Netherlands, 2010.

[74] B. Krishnamurthy and C. E. Wills. On the leakage of personally identifiable information via online social networks. In *Proceedings of ACM WOSN '09*, Barcelona, Spain, 2009.

[75] M. Kryczka, R. Cuevas, A. Cuevas, C. Guerrero, and A. Azcorra. Measuring BitTorrent ecosystem: Techniques, tips and tricks. *IEEE Communications Magazine Vol. 49, Issue 9*, 2011.

[76] M. Kryczka, R. Cuevas, R. Gonzalez, A. Cuevasa, and A. Azcorra. TorrentGuard: stopping scam and malware distribution in the BitTorrent ecosystem. *Under submission*, 2012.

[77] M. Kryczka, R. Cuevas, C. Guerrero, and A. Azcorra. Unrevealing the structure of live BitTorrent swarms: methodology and analysis. In *Proceedings of IEEE P2P'11*, Kyoto, Japan, 2011.

[78] N. Laoutaris, D. Carra, and P. Michiardi. Uplink allocation beyond choke/unchoke or how to divide and conquer best. In *Proceedings of ACM CoNEXT'08*, Madrid, Spain, 2008.

[79] V. Latora and M. Marchiori. Efficient behavior of small-world networks. *Physical Review Letters, Vol.87 No.19*, 2001.

[80] A. Legout, N. Liogkas, E. Kohler, and L. Zhang. Clustering and sharing incentives in BitTorrent systems. In *Proceedings of ACM SIGMETRICS 2007*, San Diego, USA, June, 2007.

[81] A. Legout, G. Urvoy-Keller, and P. Michiardi. Rarest first and choke algorithms are enough. In *Proceedings of ACM IMC'06*, Rio de Janeiro, Brazil, 2006.

[82] D. Levin, K. LaCurts, N. Spring, and B. Bhattacharjee. BitTorrent is an auction: analyzing and improving BitTorrent's incentives. In *Proceedings of ACM SIGCOMM'08*, Seattle, USA, 2008.

[83] J. Liang, R. Kumar, Y. Xi, and K. Ross. Pollution in P2P file sharing systems. In *Proceedings of IEEE INFOCOM'05*, Miami, USA, 2005.

[84] J. Liang, N. Naoumov, and K.W. Ross. The index poisoning attack in P2P file sharing systems. In *Proceedings of IEEE INFOCOM '06*, Barcelona, Spain, 2006.

[85] M. Lin, J.C.S. Lui, and D.-M. Chiu. An ISP-friendly file distribution protocol: analysis, design and implementation. *IEEE Transactions on Parallel and Distributed Systems*, 2009.

[86] N. Liogkas, R. Nelson, E. Kohler, and L. Zhang. Exploiting BitTorrent for fun (but not profit). In *Proceedings of IPTPS'06*, Santa Barbara, USA, 2006.

[87] N. Liogkas, R. Nelson, E. Kohler, and L. Zhang. Exploring the robustness of BitTorrent peer-to-peer content distribution systems. *Concurrency and Computation: Practice and Experience*, 2007.

[88] Z. Liu, P. Dhungel, Di Wu, C. Zhang, and K.W. Ross. Understanding and improving incentives in private P2P communities. In *Proceedings of ICDCS'10*, Genoa, Italy, 2010.

[89] T. Locher, P. Moor, S. Schmid, and R. Wattenhofer. Free riding in BitTorrent is cheap. In *Proceedings of HotNets'06*, Irvine, USA, 2006.

[90] F. J. Massey. The Kolmogorov-Smirnov test for goodness of fit. *Journal of the American Statistical Association, Vol. 46*, Vol. 46, 1951.

[91] D. Menasche, A. Rocha, B. Li, D. Towsley, and A. Venkataramani. Content availability in swarming systems: Models, measurements and bundling implications. In *Proceedings of ACM CoNEXT'09 Rome*, Rome, Italy, 2009.

[92] P. Michiardi, K. Ramachandran, and B. Sikdar. Modeling seed scheduling strategies in BitTorrent. In *Proceedings of IFIP Networking '07*, Atlanta, USA, 2007.

[93] A. Moshchuk, T. Bragin, S. Gribble, and H. Levy. A crawler-based study of spyware on the web. In *Proceedings of NDSS'06*, San Diego, USA, 2006.

[94] G. Neglia, G. Reina, H. Zhang, D. Towsley, A. Venkataramani, and J. Danaher. Availability in BitTorrent systems. In *Proceedings of IEEE INFOCOM '07*, Anchorage, USA, 2007.

[95] J. S. Otto, M. A. Sanchez, D. R. Choffnes, F. E. Bustamante, and G. Siganos. On blind mice and the elephant: understanding the network impact of a large distributed system. In *Proceedings of ACM SIGCOMM'11*, Toronto, Canada, 2011.

[96] M. Piatek, T. Isdal, T. Anderson, A. Krishnamurthy, and A. Venkataramani. Do incentives build robustness in BitTorrent? In *Proceedings of NSDI'07*, Cambridge, USA, 2007.

[97] M. Piatek, H. V. Madhyastha, J. P. John, A. Krishnamurthy, and T. Anderson. Pitfalls for ISP-friendly P2P design. *Proceedings of ACM HotNets-VIII*, 2009.

[98] F. Picconi and L. Massoulie. ISP friend or foe? making P2P live streaming ISP-aware. In *Proceedings of IEEE ICDCS'09*, Montreal, Canada, 2009.

[99] J.A. Pouwelse, P. Garbacki, D.H.J. Epema, and H.J. Sips. The BitTorrent P2P file-sharing system: Measurements and analysis. In *Proceedings of IPTPS'05*, Ithaca, USA, 2005.

[100] D. Qiu and R. Srikant. Modeling and performance analysis of BitTorrent-like peer-to-peer networks. In *Proceedings of ACM SIGCOM'04*, Portland, USA, 2004.

[101] S. Ren, L. Guo, and X. Zhang. ASAP: an AS-aware peer-relay protocol for high quality VoIP. In *Proceedings of ICDCS'06*, Lisboa, Portugal, 2006.

[102] P. Shah and J.-F. Paris. Peer-to-Peer multimedia streaming using BitTorrent. In *Proceedings of IEEE IPCCC'07*, New Orleans, USA, 2007.

[103] A. Sherman, J. Nieh, and C. Stein. Fairtorrent: Bringing fairness to Peer-to-Peer systems. In *Proceedings of ACM CoNEXT'09*, Rome, Italy, 2009.

[104] S. Shin, J. Jung, and H. Balakrishnan. Malware prevalence in the Kazaa file-sharing network. In *Proceedings of ACM IMC'06*, Rio de Janeiro, Brazil, 2006.

[105] J. Shneidman, D. Parkes, and L. Massoulie. Faithulness in internet algorithms. In *Proceedings of PINS'04*, Portland, USA, 2004.

[106] G. Siganos, J. M. Pujol, and P. Rodriguez. Monitoring the BitTorrent monitors: A bird's eye view. In *Proceedings of PAM'09*, Seoul, Korea, 2009.

[107] G. Siganos, X. Yang, and P. Rodriguez. Apollo: Remotely monitoring the BitTorrent world. Available at: `http://research.tid.es/georgos/images/apollo_imc09.pdf`, Technical Report, Telefonica Research, 2009.

[108] M. Sirivianos, J. Han, P. Rex, and C.X. Yang. Free-riding in BitTorrent networks with the large view exploit. In *Proceedings of IPTPS '07*, Bellevue, USA, 2007.

[109] M. Steiner and Biersack E. W. Crawling azureus. Available at: `http://www.eurecom.fr/~btroup/BPublished/RR-08-223.pdf`, Technical Report, Institut Eurecom, 2008.

[110] Y. Tian, D. Wu, and K.-W. Ng. Analyzing multiple file downloading in BitTorrent. In *Proceedings of ICPP '06*, Columbus, USA, 2006.

[111] G. Urvoy-Keller and P. Michiardi. Impact of inner parameters and overlay structure on the performance of BitTorrent. In *Proceedings of IEEE INFOCOM'06*, Barcelona, Spain, 2006.

[112] A. Vlavianos, M. Iliofotou, and M. Faloutsos. BiToS: Enhancing BitTorrent for supporting streaming applications. In *Proceedings of IEEE INFOCOM '06*, Barcelona, Spain, 2006.

[113] D. J. Watts and S. H. Strogatz. Collective dynamics of small-world networks. *Nature*, 1998.

[114] D. Wu, P. Dhungel, X. Hei, C. Zhang, and K.W. Ross. Understanding peer exchange in BitTorrent systems. In *Proceedings of IEEE P2P'10*, Delft, the Netherlands, 2010.

[115] H. Xie, Y. R. Yang, A. Krishnamurthy, Y. Liu, and A. Silberschatz. P4P: Provider portal for applications. In *Proceedings of ACM SIGCOM'08*, Seattle, USA, 2008.

[116] W. Yang and N. B. Abu-Ghazaleh. GPS: A general Peer-to-Peer simulator and its use for modeling BitTorrent. In *Proceedings of IEEE MASCOTS'05*, Atlanta, USA, 2005.

[117] C. Zhang, P. Dhungel, D. Wu, Z. Liu, and K.W. Ross. BitTorrent darknets. In *Proceedings of IEEE INFOCOM'10*, San Diego, USA, 2010.

[118] C. Zhang, P. Dhungel, D. Wu, and K.W. Ross. Unraveling the BitTorrent ecosystem. *IEEE Transactions on Parallel and Distributed Systems*, 2010.

[119] L. Zhou, L. Zhang, F. McSherry, N. Immorlica, M. Costa, and S. Chien. A first look at peer-to-peer worms: Threats and defenses. In *Proceedings of the IPTPS'05*, Ithaca, USA, 2005.

[120] Z. Zhu, G. Lu, Y. Chen, Z.J. Fu, P. Roberts, and Keesook Han. Computer software and applications. In *Proceedings of COMPSAC '08*, Turku, Finland, 2008.