Chapter 6
Privacy in Peer-to-Peer Networks

Diego Suárez Touceda, José María Sierra Cámara and Jesús Téllez Isaac

6.1 Introduction

In the client-server connectivity model participants' roles are clearly defined and are not interchangeable: one or more servers offer a range of services used by a group of clients. Although this model is a valid solution in most scenarios, there are environments in which the model is not feasible for technical, financial, social, or security reasons [1].

Conversely, P2P networks are designed to take advantage of dispersed network resources and enable participants to act as servers or clients (without the need for a fixed role); their main characteristics being the direct sharing of resources among users, their self-organization, stability, and autonomy. Based on this connectivity model, a multitude of applications have emerged, such as distributed computing systems Seti@home [2] or Genome@home [3], distributed database systems such as PIER [4] or Piazza [5], content distribution applications including Napster [6], Gnutella [7], or BitTorrent [8], and communications systems such as P2PSIP [9] or Skype [10].

D.S. Touceda (✉)
Evalues IT Evaluation Lab, Universidad Carlos III de Madrid,
Avda. Gregorio Peces Barba 1, 28918 Leganés Madrid, Spain
e-mail: diego.suarez@uc3m.es

J.M.S. Cámara
Computer Science Department, Universidad Carlos III de Madrid,
Avda. de la Universidad 30, 28911 Leganés Madrid, Spain
e-mail: sierra@inf.uc3m.es

J.T. Isaac
Computer Science Department, Universidad de Carabobo,
Avda. Universidad, Valencia, Venezuela
e-mail: jtellez@uc.edu.ve

S. Zeadally and M. Badra (eds.), *Privacy in a Digital, Networked World*,
Computer Communications and Networks,
DOI 10.1007/978-3-319-08470-1_6

However, despite their assets, P2P networks offer little privacy protection. Data, that can be sensitive, is no longer stored in trusted servers but in peers (potentially untrusted), and can be openly accessed and used (e.g., for advertising, users profiling and impersonation, etc.) [11]. Also, user accesses and publishing are no longer done through large companies, such as Google or Facebook, that have privacy policies protecting the users personal information from being public. These accesses are done through less-trustworthy entities that can disclose the users personal information or track its behavior [12]. Several P2P applications propose mechanisms to ensure privacy such as OceanStore [13] or Past [14]. However, these solutions remain insufficient. Managing privacy is not possible in current P2P networks without adding new privacy functionalities [11].

Therefore, users need to be concerned about privacy in P2P networks. They should know how to properly configure and use the software to restrict the information being shared. It is very common for a user to share the entire hard drive, including sensitive information, without knowing it. Hence, users have to be sure that they are not sharing personal information which could be exploited by malicious users [15].

With this problem in mind, in this chapter we analyze the existing privacy issues in P2P networks and the solutions that can be used to prevent them, aiming to help the reader to better understand privacy in P2P networks and applications.

The rest of the chapter is structured as follows. An overview of P2P networks is given in Sect. 6.2. Section 6.3 analyzes the privacy issues of P2P networks. The existing solutions that can be used to improve privacy in P2P networks are presented in Sect. 6.4. Section 6.5 discusses how the described solutions can be used to mitigate the existing issues and explore the challenges that must be addressed in the future. Finally, Sect. 6.6 outlines the conclusions of the contents presented in this chapter.

6.2 Background

In this section, we present some basic knowledge on P2P including its definition, classification, the layer architecture, and the applications.

6.2.1 P2P Definition

A review of the literature reveals that, due to the considerable number of different definitions of "peer-to-peer", there is no accurate definition of P2P today, mainly distinguished by the "broadness" they attach to the term. Many definitions are proposed trying to capture the main features of P2P systems. Some typical definitions include the following:

- A system to be P2P if the elements that form the system share their resources in order to provide the service the system has been designed to provide. The elements in the system both provide services to other elements and request services from other elements [16].
- Peer-to-peer systems are distributed systems consisting of interconnected nodes able to self-organize into network topologies with the purpose of sharing resources such as content, CPU cycles, storage and bandwidth, capable of adapting to failures and accommodating transient populations of nodes while maintaining acceptable connectivity and performance, without requiring the intermediation or support of a global centralized server or authority [17].
- A distributed network architecture may be called a Peer-to-Peer (P-to-P, P2P, …) network, if the participants share a part of their own hardware resources (processing power, storage capacity, network link capacity, printers, …). These shared resources are necessary to provide the Service and content offered by the network (e.g. file sharing or shared workspaces for collaboration). They are accessible by other peers directly, without passing intermediary entities. The participants of such a network are thus resource (Service and content) providers as well as resource (Service and content) requestors (Servent-concept) [18].

According to the above definitions, the following characteristics of P2P systems can be identified [19]:

- **Shared resources**. Peers in P2P systems are designed for sharing resources, by direct exchange, with each other in order to provide the services or content offered.
- **Self-organized**. Peers in P2P systems are self-organized into network topologies, respecting the autonomy of peers.
- **Dual role**. Peers in P2P systems act both as clients (requesting resources from others) and servers (providing resources to others) at the same time.
- **Stability**. P2P systems have the ability to adapt to peer failures (fault-tolerance) and to accommodate a large number of participating peers (scalability).
- **Autonomy**. Each peer maintains and controls, without the support of a central server or authority, its own content and resources.

Due to the absence of a centralized server, a P2P network is designed around the concept of each peer being client and server simultaneously. This model of network layout differs from the client-server model, which has a centralized server responsible for controlling the access of shared resources within the network, giving the clients limited privileges. Therefore, the P2P architecture is considered opposite of the client-server model.

Although the P2P model is perceived as an advantage, it introduces many management and security issues since there is no control over the content being shared within the network. Hence, the participating peers become prone to various threats and security violations [20]. Figure 6.1 illustrates the client-server architecture whilst Fig. 6.2 shows the P2P architecture.

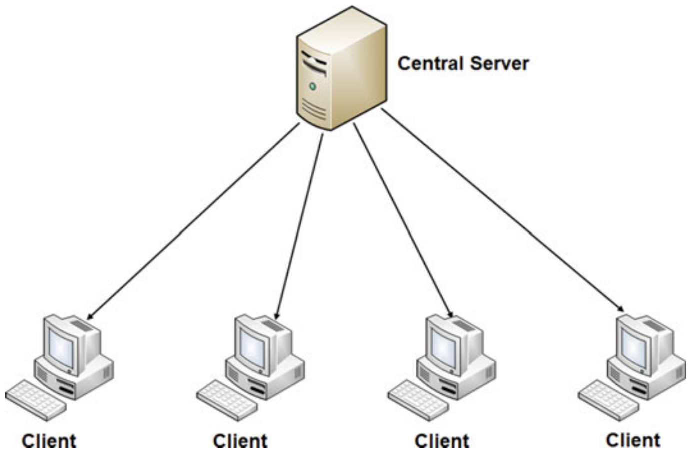

Fig. 6.1 Client-server architecture

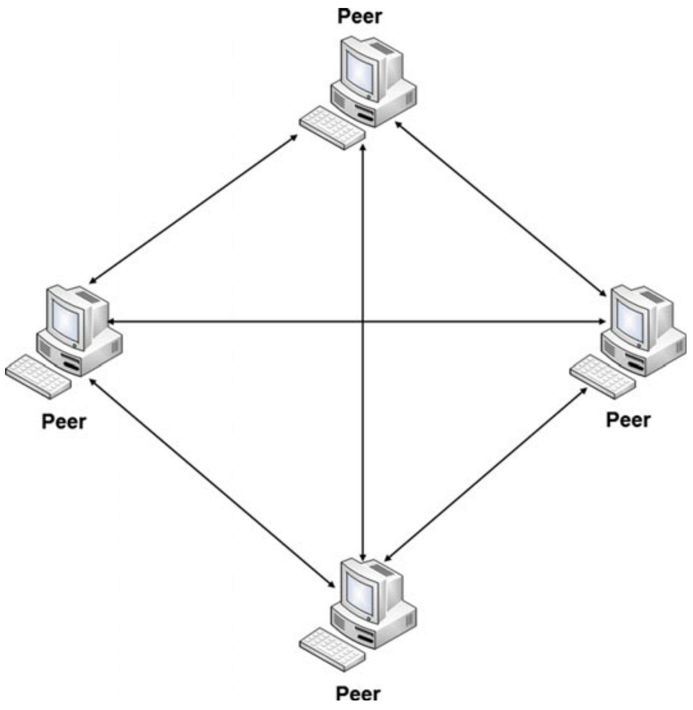

Fig. 6.2 P2P architecture

6.2.2 P2P Systems

A P2P system consists of three layers: underlying network, overlay network, and application. Figure 6.3 illustrates the model of a P2P system in a layered architecture [19].

- Underlying network is the communication network to which peers connect for routing packets.
- Overlay network is the network of peers that relies on the underlying network for routing packets to each other. It is responsible for providing P2P services for P2P applications built on top of it, performing many operations such as storing and retrieving data, management of nodes, management of resources, management of security, and so on.
- The Application-level layer is concerned with the content and service provided to users by P2P applications (such as P2P file-sharing applications, P2P instant messaging applications, P2P video streaming applications, and so on) using the P2P services provided by the overlay network.

In the following sections, we discuss the overlay network layer and the application layer of P2P systems.

6.2.3 P2P Overlays

In the last decade, several P2P overlay networks have emerged with diverse network structures, topologies, routing algorithms, and so on. Based on the network structure, P2P overlays can be classified in several types (see Fig. 6.4) that will be briefly discussed below.

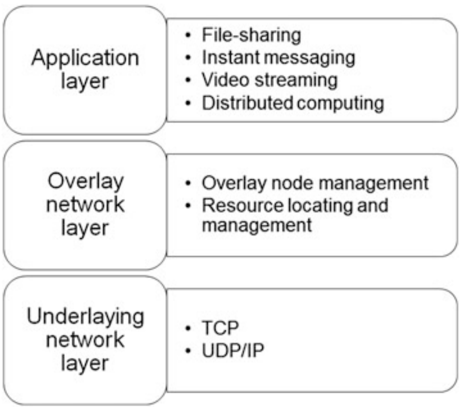

Fig. 6.3 Layer architecture of P2P systems

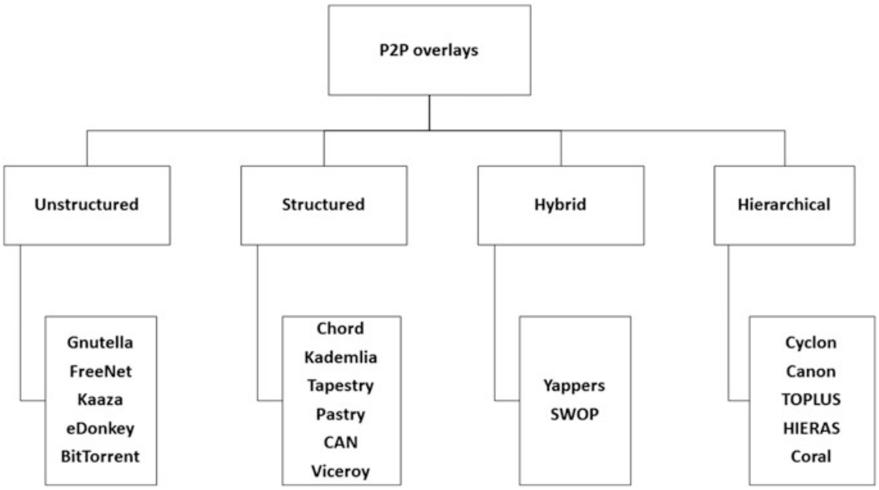

Fig. 6.4 Summary of overlay networks

6.2.4 Unstructured P2P Networks

In this category, peers are organized by the P2P overlay network into a random graph (in a flat or hierarchical manner), which means that it is not possible to establish a correlation among a peer and the content handled by it; owing to the arbitrary creation of the links between nodes. Therefore, in order to query content stored by overlay peers, an unstructured P2P network has to use flooding, random walks, or expanding-ring time to live (TTL) search, and so on on the graph. When a peer is visited, it will evaluate the query locally on its own content, and will support complex queries.

For the purpose of routing, a peer builds and maintains a local routing table (which contains some neighbor peers) by periodically checking the aliveness of these neighbors to remove the unavailable ones and to update them with new ones available. The maximum number of neighbors that a peer has is limited in an overlay to ensure the scalability.

Due to the absence of constraints about the topology of unstructured P2P overlay networks, these systems are easy to build and maintain [21]. Moreover, they support complex queries in an easy way, and they are highly robust against high rates of peers frequently joining and leaving the network (Churn) [19]. However, a limitation of unstructured P2P overlays is their scalability because the usage of flooding algorithms by peers produces a high network traffic [22, 23]. Also, because queries for content are not widely replicated and must be sent to a large fraction of peers, unstructured P2P overlays become inefficient [24].

Examples of unstructured P2P overlay networks are the following: Freenet [25] Gnutella [26, 27], FastTrack [28] /KaZaA [29], Overnet [30]/eDonkey2000 [31], and BitTorrent [32].

6.2.5 Structured P2P Networks

In this category, in contrast to unstructured P2P networks, structured P2P overlay networks provide a geometric topology that is tightly controlled and where contents are placed in specific locations and not in random peers.

A Distributed Hash Table (DHT) is implemented on most of the structured overlays based on an abstract key space. A unique key is assigned to each peer or data item which is taken from the key space using consistent hashing. To store a value in a node, a DHT must determine the node that has the minimal distance among the value's key and the node's identifier [19].

Two operations are provided by a DHT: a store operation (put(key,value)) and a retrieval operation (value = get(key)). The DHT will route the requested operation for a given key to the node responsible for the key. Each node maintains, for routing purposes, overlay links to a number of other nodes. Also, the IP address, the node's identifier, and other information of each of these nodes is stored in their routing tables. Using these routing tables, nodes forward the requests that receive to their closer link to the destination until these requests finally reach the destination node.

Structured P2P overlay networks provide a cooperative, stable, and robust mechanism for storing and retrieving content when their algorithms are executed correctly. Nevertheless, this class of systems does not support complex queries in its simple form, hence it is necessary to store a copy or a pointer to each data object (or value) at the peer responsible for the data object's key. Also, most of them deploy very minimalist security mechanisms that make them an attractive target for attackers [19, 24, 33].

Examples of structured P2P overlay networks are the following: Content Addressable Network (CAN) [34], Tapestry [35], Chord [36], Pastry [37], Viceroy [38], and Kademlia [39].

6.2.6 Hybrid P2P Networks

Hybrid P2P systems combine unstructured and structured topologies in their hierarchy with the intention of exploiting the advantages of kind of networks. These systems employ structured overlay topologies at their upper level whilst unstructured overlay topologies are used at their lower level, or vice versa.

6.2.7 Hierarchical Overlays

A hierarchical overlay is an overlay architecture that uses multiple overlays to organize its peers in a nested fashion (which are interconnected in a tree). Therefore, a message can be sent to a peer in a different overlay by forwarding the message to the nearest common parent overlay in the hierarchy with the destination peer. Examples of hierarchical overlays include: Cyclon [40], Hieras [41], Canon [42], and TOPLUS [43].

Hierarchical overlays can increase overall performance in P2P systems that exhibit locality in their operations.

6.2.8 P2P Applications

Nowadays, P2P systems have become one of the most common technologies used in the Internet. Although until 1999 the most common paradigm in the Internet was the client-server model, the emergence of Napster [44] (a P2P file-sharing application used to share music) has attracted attention and pushed P2P systems to become one of the most used (and controversial) technologies [45].

The P2P architecture has been widely used worldwide because of the following advantages [46]:

- **Scalability**. In P2P networks users contribute with resources, meaning that while the number of users increases, the ability of the network will also increase due to the additional resources brought by the new users.
- **Resilience**. P2P architecture prevents the single point of failure (inherent to client-server systems), hence, increasing the robustness and reliability of the system.
- **Cost-efficiency**. The installation and management cost are reduced in P2P networks by incorporating the resources of the users and not using dedicated servers.

Despite of the above advantages, P2P networks also have their limitations. One example are the new security threats that can emerge due to the direct exchange of resources among end-users without the participation of a secure server. Besides their most famous application, file-sharing, P2P systems have also been used for a variety of different application categories, including the following [17]:

- **Communication and collaboration**. Systems in this category provide the necessary infrastructure to allow the direct communication and collaboration between peer computers. Examples include instant messaging applications such as Aol, Yahoo, and MSN.
- **Distributed computation**. The goal of this category of systems is to take advantage of the available computer processing power (CPU cycles) of the peers of the network by decomposing a computer-intensive task into small work units which should be distributed among them. Every peer computer executes its corresponding work unit once it has been received before returning the results. Examples include projects such as Seti@home [47] and Genome@home [48].
- **Internet service support**: Many different applications have emerged based on P2P infrastructures to support a variety of Internet services, such as P2P

multicast systems [49], Internet indirection infrastructures [36], and security applications, providing protection [50].

- **Content distribution**. In this category fall most of the current P2P systems that range from relatively simple direct file-sharing applications, to more sophisticated systems which create a distributed storage medium for securely and efficiently publishing, organizing, indexing, searching, updating, and retrieving data. Within the P2P content distribution domain, applications can be grouped as follows:

  - **P2P file-sharing**: P2P is one of the most successful architectures for file-sharing, both using structured and unstructured overlays. In this kind of networks, information about shared files (frequently stored locally at the owners machines) can be delivered to some specific peers or saved in a central server. In order to download a file, a peer has to perform the following steps: searching and downloading. In the first step, a query is sent by the initiating peer to the network. Once received, this query is replied by the peers that keep files that match the search with the information about these files. However, there are also alternatives to this searching mechanism. For example, in BitTorrent [32] (a P2P system that uses a central location to manage users' downloads), contents are located using special types of files called â€œtorrent files" (usually publicly accessible in web servers) which contain information about the content file, its length, name, hashing information, and the URL of a tracker (responsible for maintaining track of all the peers storing, both partially and completely, the content file) [19, 24, 32]. In the second step (downloading), the initiator peer reach directly the peers where the file is stored in order to complete the exchange of the file.

    In Fig. 6.5, the architecture and operation of BitTorrent are illustrated. A peer P1 which host a file and acts as the seed, is responsible for the following

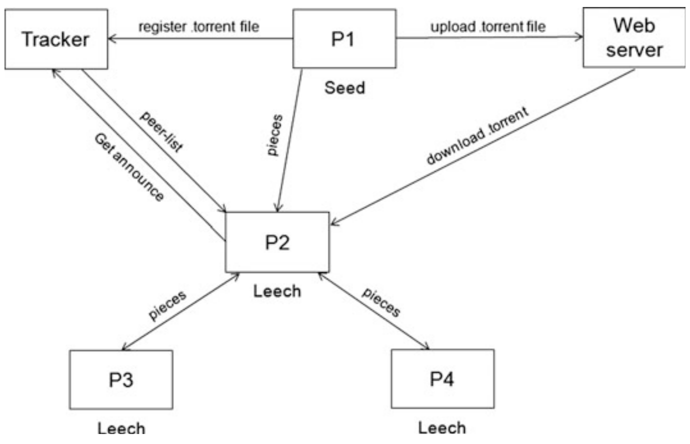

Fig. 6.5 BitTorrent architecture and operation

steps: upload the "torrent file" to a web server and register it to the tracker. The peer P2 could download the file by sending a Get message to the tracker whoever replies with a list of the peers that are hosting this file (P1, P2 and P3 in this case). Once the list is received by P2, it contacts peers P1, P3 and P4 to retrieve pieces of the file. Since P3 or P4 do not have the file completely, it can also retrieve the missing pieces from P2 (the ones P2 has retrieved from P1 and that P3 or P4 do not have) [19].

- **P2P streaming**: This category of P2P systems have been deployed recently to provide live and on-demand video streaming services on Internet at low cost. P2P streaming solutions create an overlay network topology for delivering content formed by the users of the network. These users can download or upload video assets, thus becoming active participants in the streaming process. rStream [51] is an example of this kind of application.
- **Video-on-Demand** (**VoD**): This is a technology that ensures the availability of a whole video at the time of the transfer. Since no content is generated/updated while data is transferred and rendered, users should be able to watch any point of the video at any time. In a P2P-based VoD (P2P-VoD) application, the entire stored media file needs to be retrieved at a rate that allows the recovered pieces of the file to be played in sequential order at the media playback rate. Therefore, if the retrieval rate is sufficient, the playback phase extends beyond the transfer phase. Examples include SplitStream [52] and pcVOD [53].
- **Live streaming**: In P2P live streaming applications, a root server generates video content in real time while peers connect to it as clients forming a tree. Content dissemination is acted upon by other peers (clients). The video playback is synchronized among all peers unlike peers in a VoD-like network, where each peer may be positioned in a different part of the video [45]. Anysee is an example of this kind of application [54].
- **P2PTV**: In a typical architecture of P2PTV, there are trackers that contains the information of the peers which distribute a specific channel. Therefore, to view a channel, a peer needs to query the tracker that is distributing the channel; to finally contact the peers of the trackers directly to receive the video stream from them. In these systems, each user is able to download a video stream and simultaneously upload it to other users in order to contribute to the overall available bandwidth. [19]. Examples of P2PTV application are Zattoo [55] and PPLive [56].
- **Other content distribution systems**: There are other content distribution applications developed using P2P technologies such as systems for pre-recorded TV program distribution (such as BBC-iplayer [57]), game updates, and so on [19].

6.3 Privacy Issues in P2P Networks

As noted in [58], the term privacy is an umbrella term, referring to a wide and disparate group of related things. The use of such a broad term is helpful in some contexts yet quite unhelpful in others. One of these contexts is security. In order to analyze the existing privacy issues in P2P Networks, we need to be more specific and not talk about privacy in general, but about specific users private attributes that may be under threat.

6.3.1 Privacy of User Identity

In order to connect to a P2P network users need to reveal some information about themselves (at least to the peers they are directly connected to), such as their IP address and port number. Also, if some kind of enrollment mechanism is in place, users may have to provide more information to satisfy the enrollment policy, for example, an ID, password, and so on. In some cases, a centralized mechanism, such as the offline certification authority (CA) used in RELOAD [9], can be used to control the enrollment to the network. In this way, users credentials are stored in a trusted third party (TTP) that can protect the privacy of the users attributes. However, to join the network, some information (credentials) must still be revealed to other peers (that may be malicious) to prove that the user is an authorized member of the network. Furthermore, when a user requests access to a resource stored in another peer it may need to reveal some information about itself to satisfy the resource access control policy. And, in the same way, the resource provider may need to reveal some information too, for example, for proving it is a valid content provider, to provide access to the resource.

Therefore, as privacy issues related to the users identity, we identify:

- Internet service provider (ISP) identification of peers through relating an IP address and port with the users subscription data.
- TTP identification of peers using the provided users attributes requested during the enrollment process.
- Directly connected peers identification of users through their IP address and port and (when used) the credentials needed to prove membership to the network.
- Resource owner identification of users through the credentials needed to satisfy the resources access control policy.
- User identification of resources providers through the credentials used while providing access.

The first two cases are common to most of the Internet applications: users have to reveal some private attributes to both their ISP, to contract a line to access the Internet, and to their service providers server (mail account, cloud storage, etc.), to be able to access their services. However, the last three threats are new. The first of these three cases appears due to the fact that in P2P networks users are directly connected to other (usually unknown) peers of the networks that may be malicious. The second and the third appear because users no longer access resources stored in a trusted server but in (usually unknown) peers that, again, may be malicious.

6.3.2 Privacy of User Location

As presented already, peers of a P2P network know the IP address and port of the peers they are directly connected to. Also, some applications need to make public these attributes of a user to all the participants of the network in order to allow them to communicate with each other. For example, in P2PSIP applications users IDs and locations (IP addresses and port numbers) are published in a DHT formed by all the peers of the network. This information can be used by other users of the network that wish to communicate with them to start VoIP phone calls or to send chat messages. However, as noted in [59], since this location information is stored in the DHT, the user cannot know which peers have requested it. Therefore, it is possible for malicious peers to lookup the user regularly and to map its IP addresses to geographic locations without the user being aware of it. This information can be used by those with malicious intent to make profiles of the users location. The same happens with other similar applications, such as Skype, as commented in [12].

Summarizing, the main location privacy issue for a P2P user is other peers having access to its location and being able to make a profile about its mobility.

6.3.3 Privacy of the User Access

In Sect. 6.3.1 we have already talked about the privacy of users identity when requesting access to a resource. Another related but different privacy issue appears during this access: the peer responsible for a resource also can monitor the accesses of the users of the network to it. In the same way, as noted in [15], the searched keywords and the downloaded files can be gathered. This information in conjunction with the users IP addresses can be used to create a database about the users accesses. One example of an application suffering this threat is P2PSIP. In response to this concern, the research [60] states that the P2P routing protocol opens the possibility for P2P users to record the activities of other users in the network. On the one hand, all the communications of a user are done through the fingers in its routing table that can monitor its activities. On the other hand, the peer responsible for storing a resource can monitor all the accesses over the resource it controls.

So, basically, the main threats here are users being monitored in the system by other peers; both when accessing and looking for a resource. This is also known as sender anonymity.

6.3.4 Privacy of User Publishing

Another privacy issue appears for a user publishing a resource: the party accessing or looking for a resource also learns which resources the user has published. This could be used, for example, for censorship governments to make a search for forbidden contents and ban or even prosecute the users of the network providing them. This is also known as recipient anonymity.

6.3.5 Privacy of Contents

In general, before reaching their final destination, Internet communications traverse several systems that may spy or even modify the exchanged data. This is even more notorious in P2P networks, where users rely on other peers to access the systems resources, providing the possibility for an intermediate peer to monitor the contents of a users communication.

Also, unlike the client-server model where data is centrally stored, in P2P contents are spread among all the peers of the network; being each peer responsible for a specific part of it. These contents can be public information, available for other users of the network, or non-public data, only available for the users private access. Due to the fact that the peer responsible for storing these contents may be malicious, a security mechanism should be implemented in order to prevent the storing peer or an attacker from accessing this data without authorization [60].

6.3.6 Application Misconfiguration and Misuse

In P2P networks computers act simultaneously as clients and servers. However, as noted in [15], to properly configure a server is not an easy task that typical end-users can do. To configure a secure Internet server requires professional experience. Therefore, despite configuring and using P2P software being easy, using it in a controlled way is much more complicated. As an example, in [61] a study about Kazaa usability is presented showing that it is very common for users to share private data: emails, text documents, configuration files, or even the whole disk without being aware of it. Since a user would not do this on purpose, it seems clear that we are facing an application misconfiguration problem.

Application misconfiguration therefore clearly represents another threat to privacy that should be taken into account.

6.3.7 Spyware and Malware

Spyware and Malware are further privacy threats for P2P applications. Spyware spies on the user by collecting information about its activities without the user's knowledge. For its part, malware goes a step further performing more harmful activities such as collecting more sensible information (credit card numbers, passwords, etc.) or giving remote control to an attacker over the users device.

Several P2P applications are bundled with spyware that gather and send information about the users activities to a third party. In turn, that third party uses the gathered data to gain information about their potential customers [15]. However, it is hard to check the spying activities and information leaked by these applications because most of them are closed-source. One example of such an application that includes spyware is Kazaa.

Other related threats to users privacy are the programs downloaded from P2P networks. Since the sources in a P2P network may not be trustworthy, these programs can contain spyware or malware that compromise the privacy of the user.

6.4 Solutions for Privacy Issues in P2P Networks

Most communications over the Internet lack privacy protection: the messages are not encrypted nor are the sender and the receivers identities protected. It is really difficult to achieve full protection against privacy threats on the Internet, either P2P based or not, just as normal people do not have the necessary means to be fully protected against professional thieves in the physical world [62]. However, average protection for the vast majority of scenarios is definitely possible. In the rest of this section, we analyze different solutions that could help a user to achieve a desirable level of privacy.

6.4.1 Anonymous Systems

Anonymity can enable censorship resistance and freedom of behavior without fear of persecution. Anonymity is mostly used to hide user identity. If anonymous communication channels are used, a channel listener is not able to understand the messages sent on the channel or who has sent them [11]. Since one of the first papers on anonymity was published [63] by D. Chaum in 1981 outlining an electronic mail system to hide who a participant communicates with through the use of mixes (nodes hiding the correspondences between their input and output messages in a cryptographically strong way), several protocols and applications to protect anonymity have emerged [60]. However, most of them follow a similar process: with the idea of preserving the privacy of the identity of the sender and the receiver of a message, as described in [64]: â€œWhen a sender Alice sends a message to a receiver Bob via a message router Rob, she first encrypts the message under Bob's public key and then encrypts the results together with Bob's name under Rob's public key. She then sends this final encryption to Rob who decrypts it, sees another encryption plus Bob's name and thus forwards this encryption to Bob. Upon receiving it, Bob decrypts and then can read the message from Alice. Thus, from the communication packets sent between Alice, Rob, and Bob one can only tell that Alice sent some message to Rob and that Rob sent some message to Bob".

Using this initial concept, several different approaches appear to improve anonymity: cache and mix messages before they are sent, use several routers in a row to increase the probability that at least one of them keeps the relation between the input and the output node secret, or randomly choose at each routing hop whether the message is sent to the final destination or to another intermediate routing hop. Examples of protocols to do so are Mix-networks, Mix-cascades, and Onion Routing.

What follows is a brief overview of the more relevant anonymous systems, focusing on those specifically designed for P2P networks:

- Tarzan [65] is an anonymous P2P network. A peer that wants to send a message through the network, instead of sending it directly, creates an encrypted tunnel to another peer and asks that peer to forward the message in its behalf. This process is repeated several times, creating an onion encrypted connection, that relays the message through a succession of intermediate peers.
- MorphMix [66] is very similar to Tarzan. The main difference between them is how the route a communication follows through the network is chosen. In Tarzan, the route is chosen by the source, while in MorphMix the intermediate nodes determine the next step.
- SwarmScreen [67] is a privacy preserving layer for P2P applications that tries to obfuscate the user's network behavior. Since a users behavior can be deduced by his or her interests, SwarmScreen connects the user to other users outside of their community of interest (by adding some percentage of extra random connections that are indistinguishable from the real ones), which can disguise the users interests and thus their behavior.
- Pr2-P2PSIP [59] has been created with the idea of providing P2PSIP user registration and session establishment, while preserving the privacy of the network participants. It is based on a central authentication server (AS). After the user authenticates with the AS, the AS provides the user with a certificate that binds the identity of the user with its public key. Besides its identity, each user has two pseudonyms fi and si (temporal identities that cannot be linked with the identity of the user). These pseudonyms allow the user to participate in two different overlays: one used to store the users contact information and another for forwarding messages.

All of the summarized systems have their advantages and drawbacks (this discussion is beyond the scope of this chapter). However, all of them have two main characteristics in common: they improve the anonymity of the user, but at a high performance cost.

6.4.2 Routing Modifications

Using an anonymous system, such as the ones presented in Sect. 6.4.1, may have an appreciable impact on the systems performance. In some cases, when performance is a key factor, another mechanism, such as routing modifications, can be used in order to improve the users privacy while maintaining a good performance. Router modifications can be carried out in two different ways: varying the routing algorithm and obfuscating routing headers. Both mechanisms help to improve a users privacy, mainly in structured P2P networks where routing is more deterministic and can reveal more information about the users behavior. The rest of this section is based on the authors previous research presented in [60].

6.4.2.1 Routing Variations

The study [68] analyzes the anonymity of the Chord protocol and concludes that the implemented recursive routing algorithm provides a high degree of sender anonymity against passive observers.

The proposal described in [69] goes a bit further and compares the anonymity of different alternatives with the original recursive routing algorithm:

- **Random recursive routing**. In this variation, peers forward the message at random to whatever finger is closer to the destination, instead of routing messages to the finger closest to its destination. Random recursive routing improves anonymity, but unfortunately it also increases the path length.
- **Weighted random routing**. Instead of picking the next forwarding hop at random from the closer fingers, fingers are weighted and picked with different probabilities: for example 1/2 for the closest, 1/4 for the second closest, and so on. In comparison to the random one, it reduces the average path length while maintaining a degree of anonymity that is nearly as good.
- **Indirect routing**. In this routing algorithm, when a peer wants to send a message, instead of routing it directly to its destination, it chooses at random an intermediary peer in the network to route the message on its behalf. Also, the query to the intermediary is secured using an m-of-n secret sharing scheme, that is, the message is split into n shares sent using independent routes and at least m shares (of n sent) need to be captured in order to reconstruct the message. This way, it is very difficult for an attacker to know the true destination of the query. Indirect routing improves the anonymity of the sender but it also increases the number of messages needed to route a query and its latency. A similar routing alternative is used in the AP3 system [70].

6.4.2.2 Headers Obfuscation

Besides the routing algorithm used, it could be also helpful, in order to improve users privacy, not to use header fields that may reveal information about the route followed by a message:

- Avoid setting fixed default values for TTL counters. Alternative methods, such as those implemented in Freenet [71], may be used.
- Methods such as the forwarding tables in AP3 [70] or the Truncated Via-Lists in RELOAD [9] should be used to obfuscate the information needed to route back the response of a query.

6.4.3 Protection of Contents in Transit

The most widely accepted measure to obfuscate the content of a message is encryption. However, as the authors presented in previous work [60], the special properties of routing in P2P networks suggest a clarification of how this mechanism should be implemented. In typical Internet routing, security is normally ensured end-to-end, that is, the sender encrypts and signs the message and delivers it to the recipient using some specific protocol for this task, such as TLS [72], DTLS [73], or IPsec [74]. Unfortunately, this approach is not valid in P2P networks, because the intermediate hops need access to some information of the message in order to route it properly. Therefore, the routing protocol needs to implement two features: it must separate the routing information (needed for the intermediate nodes to route the message) from the content of the message per se (that must be only accessed by the addressee). Also, it must permit use of both hop-by-hop and end-to-end security. First the sender encrypts and signs the content of the message with the public key of the ultimate receiver and the sender's private key respectively (end-to-end security at the application layer), then the sender appends to it the routing information and encrypts and signs the whole message for the first hop using TLS, DTLS, or IPsec (hop-by-hop security at the network/transport layer). This way, every hop can check and modify the routing information of the query in order to properly route it, but only the receiver can see its content. An example of a P2P protocol implementing both features is RELOAD [9].

6.4.4 Protection of Contents at Rest

Following the security analysis conducted in [60], there are mainly three mechanisms to protect the privacy of contents at rest in P2P networks: local control, cryptography, and dedicated security services.

6.4.4.1 Local Control

In this mechanism the node responsible for the resource is in charge of its access control and, therefore, of its privacy protection. Local control can be implemented in different ways. In the RELOAD protocol [9], for example, each resource identifier may contain multiple kinds of data identified by a Kind-ID. The definition of each data kind specifies rules for determining which certificates can access each Resource-ID and Kind-ID pair, controlling the data access. Another possibility is to use an access control list (ACL) to determine the privileges of each user over an object like in OceanStore [13] or Fairsite [75].

The main drawback of this approach is that, if the node responsible for the resource is malicious, it can access the resources content or allow unauthorized users to access it.

6.4.4.2 Cryptography

The more effective way to prevent malicious users from accessing the private data of other users within the network is to use cryptography. If a user wants to store a private resource for personal use, symmetric cryptography, such as the AES algorithm [76], could be used to encrypt the data before storing them in the network. On the other hand, if the private resource is intended to be accessed by other users, such as a voice-mail, it may be encrypted using the public key of the recipient, as described in [77]. If the publisher wants the resource to be accessible by a group of users, three possibilities arise: (1) to extend the scenario of a single recipient by storing one copy of the resource for each recipient encrypted with his or her corresponding public key; (2) to encrypt the symmetric key using the public key of all authorized readers and store the encrypted keys with the resource [75], or (3) to store only one copy of the resource encrypted with a symmetric key and send a private message to each recipient with the location of the resource in the network and the key needed to access it [13].

6.4.4.3 Dedicated Security Servers

Another solution is to add dedicated security servers to the architecture. The users rely on these servers for storing their private resources. Unfortunately, as noted in [78], these servers reduce the advantage of the P2P architecture by introducing an extra cost in its development, and issues such as load balancing and capacity problems.

6.4.5 Private and Split Credentials

Only a small number of elementary tasks can be carried out completely in an anonymous way. Usually, the user has to perform some kind of authentication, therefore revealing some private information [64]. As stated in [79], in P2P applications with anonymous authentication, if the privacy of peers is increased, the difficulties of ensuring authenticity and security are increased too. There is a clear trade-off between authentication and anonymity that is to be catered by P2P application developers.

A way to protect users access is to give users fake identities. Fake identities can be ensured by smartcard techniques where the real identity of the user is only known by the authority which distributes the smartcards. In this case, the authority must be considered as a TTP [11]. For example, in Past [14], smartcards are used to allow users to obtain necessary credentials to access resources in an anonymous fashion. However, the use of trusted party for authentication can be risky. Thus, there is a trade-off between accountability and trusted party for authentication. Users have to be able to manage their identities in order to reduce to the minimum the information about them disclosed during their operations in the network.

Private credentials [80] provide the same functionalities and security guarantees as the classical X.509 certificates, but give the user the possibility of controlling and separating its different identities. They are based on a very similar approach to the X.509 certificates, but with two particular features:

- **Single secret key, many public keys** (**cryptographic pseudonyms**). Users can create several public keys related to their secret key, instead of having only one public key for each private key. Also, it is not possible to link these public keys to each other, being unfeasible to know if they are held by several different users or by the same user.
- **Transformable**. Credentials are linked to the users secret key not to their public key. In this way, a credential related to one public key of the user can be transformed into a credential related to another and different users public key. Furthermore, the new created credential may only include a chosen subset of the attributes includes in the original credential.

Private credentials “provide the same level of security (as classical certificates), but additionally guarantee privacy during the process”, as stated in [62].

Another certificate variation that can be used together with private credentials is split certification, as presented in the authors previous work [81]. Split certification separates the identity of the user from the identity of the device the user is operating from. This way, overlay maintenance and routing communications are performed between nodes without the unnecessary knowledge of which users are connected to them. Likewise, user operations are not linked to its devices, so users perform actions in the network without having to explicitly announce the node they are operating from. Combining split certification with private credentials can reduce the users provided information when fulfilling an access control policy necessary to access a resource.

6.4.6 Hidden Services

We have already introduced (Sect. 6.4.2.1) the possibility of using an intermediate peer to route messages on a users behalf in order to protect the senders anonymity. A similar approach for P2P data sharing, presented in [82], uses buddies (community members the user has established trust relationships with) as "proxies" during data requesting. The mechanism used to hide the identity of the requester is, rather than sending the request by itself, the requester asks one or several of its buddies to look up the data on its behalf. This process can be improved by having the supplier respond to a request via its own buddies; protecting, therefore, not only the identity of the requester but also the identity of the supplier.

In this vein, but using a more sophisticated and privacy preserving approach, are the hidden services used in Tor [83]. Location hidden services allow a user to offer a service without revealing its IP address. Using Tor "rendezvous points", other users can connect to these hidden services, each without knowing the other's identity. The steps to do so are:

1. The provider picks some introduction points (peers acting as intermediates) and builds circuits (hop-by-hop encrypted random path) to them.
2. The provider publishes the service using a hidden service descriptor (including a summary of each introduction point) in a distributed hash Table
3. The requester learns about the service from its hidden service descriptor and sets up a circuit with a rendezvous point (peer acting as intermediate for the requester).
4. The requester sends a message to one of the introduction points requesting access to the service and including its rendezvous point.
5. Finally, the provider creates a circuit to the requesters rendezvous point and provides the service.

6.4.7 Application Configuration

The default configuration of a P2P application should be as strict as possible, so that the user willing to share a file had configured the application for doing so. Unfortunately, as noted in [15], this is not desirable for P2P developers. They fear that only a few users will change the configuration to allow sharing, if the default settings of the application are no sharing, therefore, reducing the interesting content available in the P2P network and forcing the users to use a different application with more content.

With this in mind, one of the most simple and useful solutions a user can implement in order to protect its privacy is to properly configuring a P2P application to only share the information the user wants before starting to use it.

6.4.8 Application Hardening

Users should use trusted applications, either open-source where the code can be checked for malicious behavior or, at least, downloaded from trusted sources, to prevent spyware or malware being installed on its computer. Also, P2P applications need to have some privileges in order to be able to carry out their activities [84]: network access, write and read permissions over the hard disk, and so on, that can be used maliciously to disclose private information or install malware on the users machine. To remedy this, [85] recommends to run the P2P application in a sandbox that isolates the portion of the hard disk the application has access to and restricts the operations it can perform. In the same way, the integrity and behavior of the whole users devices system should be checked using anti-virus and anti-malware solutions.

6.5 Recommendations, Challenges, and Opportunities

In Sect. 6.3 we presented P2P privacy issues and in Sect. 6.4 we discussed P2P privacy solutions. It is time now to see how the described solutions can be used to mitigate the existing issues and explore the challenges that must be addressed in the future, along with opportunities for new research directions. Table 6.1 presents a summary of the recommendations described during the rest of this section to protect privacy in P2P networks.

6.5.1 User Identity

As we have described already, there are five main issues related to the user identity privacy in P2P networks, two of them being common to any Internet-based system and the three others being unique to P2P networks due to the special conditions they present.

For the first issue, related to the ISP identification of peers, two possible solutions arise. The first one is to accept that in order to contract an Internet line the user should give some private information to the Internet provider and trust that it will not release it to third parties compromising the users privacy. When this solution is not acceptable, a second option is to use a public Internet access point, such as the free wireless access point provided in some public places like coffee shops, train stations, or airports.

For the second issue, related to the P2P application providers TTP identification of peers, three possible solutions arise. The first one is, again, to trust that the provider will not release the users identity information to other third parties. The second one, when possible (users are not requested to give personal identifiable

Table 6.1 P2P Privacy issues and possible solutions

P2P Privacy Issues and solutions	
Privacy issues	Solutions
User identity	Trust internet provider
	Use public access point
	Trust P2P application provider
	Use pseudonymous
	Use anonymous access applications
	Private and split credentials
User location	Privacy of user identity solutions
	Anonymous systems
	Hidden services
User access	Anonymous systems
	Routing modifications
	Protection of contents in transit
	Privacy of user identity solutions
	Private and split credentials
User publishing	Anonymous systems
	Hidden services
	Privacy of user identity solutions
	Protection of content at rest
Contents	Protection of contents in transit
	Protection of contents at rest
Application misconfiguration and misuse	Application configuration
Spyware and malware	Application hardening

information such as an account or credit card number but less identifiable information such as an e-mail that can be from an open provider), is to connect to the application using a pseudonym that cannot be related to the users real identity. Finally, the third solution is to only use P2P applications that accept anonymous users and that do not need the user to follow an enrollment process.

For the third issue, related to the directly connected peers identification of users, two possibilities arise. The first one is to trust that the Internet provider will not reveal to the other peers which user is behind the visible IP address used. The second is, again, to use public Internet access points so that the IP address can not be related to the users identity. Also, in both cases, when some kind of credential is needed in order to prove membership of the network, private and split credentials should be used.

For the fourth and fifth issues, related to the mutual identification of the resources requester and provider, the best solution is to use private and split credentials in order to hide their real identities.

6.5.2 User Location

It is impossible for a peer to hide its IP address, and therefore its location, from the peers it is directly connected to. It is, however, possible for a user to hide its location if the peers the user is directly connected to do not know the identity of the user behind the peer it is using. In order to do so, the user has to use one of the solutions presented in the previous point to hide its identity. But, what happens when the user necessarily has to make some attributes public (like an ID and IP address in P2PSIP) to allow the other users of the system to communicate with it. In this case, the best option for the user seems to be accessing the applications through an anonymous system and using a hidden service to hide its real location from the users trying to contact them. Unfortunately, the hops introduced in this communication may not be viable for a real-time communication such as P2PSIP needs. The challenge here would be to find a system that could hide the location of the user while providing it with viable connection for real-time communication. One possible approach would be to hide the real location of the user (through a hidden service) until it accepts the communication with the calling user and, once accepted, to reveal its real location to establish a direct communication valid for real-time applications. This would not hide the location of the user from an accepted caller but, at least, it would prevent a malicious user from building a profile of the user location without the user realizing it; since the real location of the user is not released until the call is accepted. However, this is still an open area of research.

6.5.3 User Access

The best way to keep private the resources accessed by a user is to use an anonymous system. This would prevent the peers directly connected to the user from knowing which resources the user requests and, also, it would prevent those responsible for a specific resource from knowing the origin of a request. Unfortunately, in some cases this solution may not be available or perhaps too costly. In such cases, a less secure option, but one that can provide some level of privacy, is to combine routing modifications with protection of the contents in transit. Another possible option is not trying to hide the access itself but the identity of the user that is behind. In this case, the user should use the solutions already described to retain the privacy of the user identity. In any case, if the access to the resource is protected by an access control policy the user should use private and split credentials in order keep the privacy of its access.

6.5.4 User Publishing

Protection against content censorship mechanism is best achieved using an anonymous system and publishing resources under a hidden service. Obtaining this protection when this solution is not possible is an ongoing problem. One possibility is to publish the content anonymously and hide the identity of the publisher using one of the solutions already described in order to protect the user identity privacy. Another possibility would be to try to implement the proxy solution, described in Sect. 6.4.6, for content distribution. One final option would be to use cryptography to protect the content of the resource at rest. Nevertheless, the last presented solution would only hide the contents of the publishing user from non authorized users. Furthermore, if one of the authorized users disclosed the content, it would be possible to prove that the user published it if the encryption for the disclosed content matched the published one.

6.5.5 Contents

In order to protect the privacy of content, both solutions for the protection of content in transit and at rest should be used. For the case of protection at rest, cryptography is the recommend solution, because it not only prevents an unauthorized user from seeing content, but it also prevents a malicious container from disclosing users private information stored in the P2P network.

6.5.6 Application Misconfiguration and Misuse

So far, we have seen how to protect the privacy of different users' attributes: its identity, location, accessed information, published resources, and private stored information. However, all these mechanisms are useless if the application used by the user to access the P2P network is not well configured and is sharing its personal private information. It is, therefore, crucial to do a proper and secure configuration of the application used before accessing a P2P network.

6.5.7 Spyware and Malware

Similarly to application misconfiguration and misuse, having spyware or malware that leaks the user's personal information can render all the privacy protecting mechanisms described before useless. So, it is very important to check that the user device is clean of spyware and malware using an application hardening solution before accessing a P2P network in order to protect the users privacy.

6.6 Conclusions

As we have seen, there are several privacy issues that must be taken into account when using P2P applications: privacy of user's identity, location, access, and publishing; privacy of contents; application misconfiguration and misuse; and spyware and malware. Also, several solutions exist that can be used to try to address these issues: anonymous systems, routing modifications, protection of contents in transit and at rest, private and split credentials, hidden services, and application configuration and hardening.

However, the choice of the one to be used in each case is a complicated task. Some solutions may be effective to protect the privacy of some attributes but ineffective in protecting others. Furthermore, implementing some of them (e.g., using an anonymous system) while neglecting others (e.g., application misconfiguration sharing documents with personal identifiable information) may render useless the solutions in place. It is, therefore, of great importance to have a holistic view of the existing issues and solutions in order to protect privacy in P2P networks.

References

1. Bryan DA, Lowekamp BB, Jennings C (2005) SOSIMPLE: A Serverless, Standards-based, P2P SIP Communication System. In: Proceedings of the First international workshop on advanced architectures and algorithms for internet delivery and applications, Washington, DC, USA. IEEE Computer Society, pp 42–49
2. The seti@home project website (1997) http://setiathome.berkeley.edu
3. The genome@home Project Website (2000) http://genomeathome.stanford.edu
4. HueBsch R, Hellerstein J, Lanham N, Thau Loo B (2003) Querying the internet with pier. In: Proceedings of the 29th VLDB conference
5. Halevy A, Ives Z, Mork P, Tatarinov I (2003) Piazza: data management infrastructure for semantic web applications. In: Proceedings of the 12th international conference on world wide web, pp 556–567
6. The Napster Website (2003) http://free.napster.com/
7. Gnutella A Protocol for a Revolution (2000) http://rfc-gnutella.sourceforge.net/index.html
8. Cohen B (2003) Incentives build robustness in BitTorrent. In: Proceedings of the 1st workshop on economics of peer-to-peer systems, P2PECON'03
9. Jennings C, Lowekamp B, Rescorla E, Baset S, Schulzrinne H (2011) Internet-draft: resource location and discovery (RELOAD) base protocol. draft-ietf-p2psip-base-13 (work in progress)
10. Skype Official Website (2003) http://www.skype.com, 2003
11. Jawad M, Serrano-Alvarado P, Valduriez P (2013) Supporting data privacy in p2p systems. In: Chbeir R, Al Bouna B (eds) Security and privacy preserving in social networks. Lecture notes in social networks, pp 195–244. Springer Vienna
12. Le Blond S, Zhang C, Legout A, Ross K, Dabbous W (2011) I know where you are and what you are sharing: Exploiting p2p communications to invade users' privacy. In: Proceedings of the 2011 ACM SIGCOMM conference on internet measurement conference, IMC '11, pp 45–60, New York, NY, USA. ACM
13. Kubiatowicz J, Bindel D, Chen Y, Czerwinski S, Eaton P, Geels D, Gummadi R, Rhea S, Weatherspoon H, Weimer W, Wells C, Zhao B (2000) OceanStore: an architecture for global-scale persistent storage. ACM SIGPLAN Notices 35(11):190–201

14. Druschel P, Rowstron A (2001) PAST: a large-scale, persistent peer-to-peer storage utility. In: Proceedings of the Eighth workshop on hot topics in operating systems, pp 75–80, Washington, DC, USA, 2001. IEEE Computer Society
15. Suvanto M (2005) Privacy in peer-to-peer networks. In: Helsinki University of Technology T-110.551 Seminar on Internetworking
16. Camarillo G (2009) Peer-to-peer (p2p) architecture: definition, taxonomies, examples, and applicability. Request for comments
17. Androutsellis-Theotokis S, Spinellis D (2004) A survey of peer-to-peer content distribution technologies. ACM Comput Surv (CSUR) 36(4):335–371
18. Schollmeier R (2001) Definition of peer-to-peer networking for the classification of peer-to-peer architectures and applications. In: Proceedings of the first international conference on peer-to-peer computing, pp 101–102
19. Ngo HG (2013) From Inter-connecting P2P overlays to co-operating P2P systems. PhD thesis, University of Nice—Sophia Antipolis
20. Bashir A (2012) Classifying p2p activities in netflow records: a case study (bittornet & skype). Master's thesis, Carleton University, Ottawa, Ontario
21. Chervenak A, Bharathi S (2008) Peer-to-peer approaches to grid resource discovery. In: Proceedings of the CoreGRID workshop on programming models grid and P2P system architecture grid systems, tools and environments, pp 59–76
22. Ripeanu M, Foster I, Iamnitchi A (2002) Mapping the gnutella network: properties of large-scale peer-to-peer systems and implications for system design. IEEE Internet Comput J 6
23. Markatos EP (2002) Tracing a large-scale peer to peer system: an hour in the life of gnutella. In: The second international symposium on cluster computing and the grid, pp 65–70
24. Lua EK, Crowcroft J, Pias M (2005) A survey and comparison of peer-to-peer overlay network schemes. IEEE Commun Surv Tutor 7(2):72–93
25. Clarke I, Miller SG, Hong TW, Sandberg O, Wiley B (2002) Protecting free expression online with freenet. IEEE Internet Comput 6(1):40–49
26. Ripeanu M (2001) Peer-to-peer architecture case study: Gnutella network. In: Proceedings of the first international conference on peer-to-peer computing (P2P 2001), pp 99–100
27. Klingberg T, Manfredi R (2002) Gnutella 0.6. http://rfc-gnutella.sourceforge.net/src/rfc-0_6-draft.html
28. Liang J, Kumar R, Ross KW (2006) The fasttrack overlay: a measurement study. Comput Netw 50(6):842–858
29. Liang J, Kumar R, Ross KW (2004) Understanding kazaa. http://infosec.pku.edu.cn/p2p/slides/2004
30. Bhagwan R, Savage S, Voelker GM (2003) Understanding availability. In: Peer-to-peer systems II. Springer, Berlin, pp 256–267
31. Heckmann O, Bock A (2002) The edonkey 2000 protocol, kom technical report 08/2002, ver. 0.8, dec. 2002. Technical report, Darmstadt University of Technology
32. Pouwelse J, Garbacki P, Epema D, Sips H (2005) The bittorrent p2p file-sharing system: measurements and analysis. In: 4th international workshop on peer-to-peer systems (IPTPS 2005), pp 205–216
33. Trifa Zied, Khemakhem Maher (2012) Taxonomy of structured p2p overlay networks security attacks. World Acad Sci Eng Technol 6(4):469–475
34. Ratnasamy S, Francis P, Handley M, Karp R, Shenker S (2001) A scalable content-addressable network. In: Proceedings of the 2001 conference on applications, technologies, architectures, and protocols for computer communications (SIGCOMM 2001), pp 161–172
35. Zhao BY, Kubiatowicz J, Joseph AD (2001) Tapestry: an infrastructure for fault-resilient wide-area location and routing. Technical Report UCB/CSD-01-1141, Computer Science Division (EECS), University of California, Berkeley
36. Stoica I, Morris R, Karger D, Kaashoek MF, Balakrishnan H (2001) Chord: a scalable peer-to-peer lookup protocol for internet applications. In: Proceedings of the 2001 conference

on applications, technologies, architectures, and protocols for computer communications (SIGCOMM 2001), pp 149–160
37. Rowstron A, Druschel P (2001) Pastry: scalable, distributed object location and routing for large-scale peer-to-peer systems. In: Proceedings of the IFIP/ACM international conference on distributed systems platforms, Heidelberg (Middleware 2001), pp 329–350
38. Malkhi D, Naor M, Ratajczak D (2002) Viceroy: A scalable and dynamic emulation of the butterfly. In: Proceedings of the twenty-first annual symposium on principles of distributed computing (PODC 2002), pp 183–192
39. Maymounkov P, Mazières D (2002) Kademlia: A peer-to-peer information system based on the xor metric. In: First international workshop peer-to-peer systems (IPTPS 2002), pp 53–65
40. Sanchez Artigas M, Garcia Lopez P, Pujol Ahullo J, Gomez Skarmeta AF (2005) Cyclone: a novel design schema for hierarchical dhts. In: Proceedings of the Fifth IEEE international conference on peer-to-peer computing (P2P 2005), pp 49–56
41. Xu Z, Min R, Hu Y (2003) Hieras: a dht based hierarchical p2p routing algorithm. In: Proceedings. 2003 international conference on parallel processing, pp 187–194
42. Ganesan P, Gummadi K, Garcia-Molina H (2004) Canon in g major: designing dhts with hierarchical structure. In: IEEE international conference on distributed computing systems (ICDCS 2004), pp 263–272
43. Garces-Erice L, Ross KW, Biersack E, Felber PA, Urvoy-Keller G (2003) Toplus: topology centric lookup service. In: Fifth international workshop on networked group communications (NGC 2003), pp 58–69
44. Saroiu S, Gummadi KP, Gribble SD (2003) Measuring and analyzing the characteristics of napster and gnutella hosts. J Multim Syst 9(2):170–184
45. Deaconescu R (2011) Protocol measurements and improvements in peer-to-peer systems. PhD thesis, University POLITEHNICA of Bucharest
46. Babaoglu Ö (2012) Introduction to peer-to-peer systems. Complex Syst Universitã 1/2 di Bologna 12:7
47. Anderson DP, Cobb J, Korpela E, Lebofsky M, Werthimer D (2002) Seti@home: an experiment in public-resource computing. Commun ACM 45(11):56–61
48. Larson SM, Snow CD, Shirts M, Pande VS (2002) Folding@home and genome@home: using distributed computing to tackle previously intractable problems in computational biology
49. van Renesse R, Birman K, Bozdog A, Dumitriu D, Singh M, Vogels W (2003) Heterogeneity-aware peer-to-peer multicast. In: Proceedings of the 17th international symposium on distributed computing (DISC2003)
50. Keromytis AD, Misra V, Rubenstein D (2002) Sos: secure overlay services. In: Proceedings of the 2002 conference on Applications, technologies, architectures, and protocols for computer communications (SIGCOMM 2002), pp 61–72
51. Wu C, Li B (2008) rstream: Resilient and optimal peer-to-peer streaming with rateless codes. IEEE Trans Parallel Distrib Syst 19(1):77–92
52. Castro M, Druschel P, Kermarrec A-M, Nandi A, Rowstron A, Singh A (2003) Splitstream: high-bandwidth multicast in cooperative environments. In: Proceedings of the nineteenth ACM symposium on operating systems principles (SOSP 2003), pp 298–313
53. Ying L, Basu A (2005) cvod: internet peer-to-peer video-on-demand with storage caching on peers. In: 11th international conference on distributed multimedia systems (DMS 2005), pp 218–223
54. Liao X, Jin H, Liu Y, Ni LM, Deng D (2006) Anysee: peer-to-peer live streaming. In: 25th IEEE international conference on computer communications (INFOCOM 2006)
55. Chang H, Jamin S, Wang W (2009) Live streaming performance of the zattoo network. In: Proceedings of the 9th ACM SIGCOMM conference on internet measurement conference (IMC 2009), pp 417–429
56. Hei X, Liang C, Liang J, Liu Y, Ross KW (2006) Insights into pplive: a measurement study of a large-scale p2p iptv system. In: Workshop on Internet Protocol TV (IPTV) services over World Wide Web in conjunction with WWW2006

57. Carlsson N, Eager DL, Mahanti A (2009) Peer-assisted on-demand video streaming with selfish peers. In: Proceedings of the 8th international IFIP-TC 6 networking conference (NETWORKING 2009), pp 586–599
58. Solove DJ (2006) A taxonomy of privacy. Technical report, 154 U Pa L Rev 477
59. Fessi A, Evans N, Niedermayer H, Holz R (2010) Pr2-P2PSIP: privacy preserving P2P signaling for VoIP and IM. In: Principles, systems and applications of IP telecommunications, IPTComm '10, New York, NY, USA. ACM, pp 134–145
60. Touceda D, Sierra JM, Izquierdo A, Schulzrinne H (2012) Survey of attacks and defenses on P2PSIP communications. IEEE Commun Surv Tutor 14(3):750–783
61. Good N, Krekelberg A (2003) Usability and privacy: a study of kazaa p2p file-sharing. In: Cockton G, Korhonen P (eds) CHI. pp 137–144. ACM
62. Camenisch Jan (2012) Information privacy?! Comput Netw 56(18):3834–3848
63. Chaum D (1981) Communications of the ACM. In: Rivest R, Chaum DL (eds) Untraceable electronic mail, return addresses, and digital pseudonyms. Commun ACM 24:84–90
64. Sandhu R, Zhang X (2005) Peer-to-peer access control architecture using trusted computing technology. In: Proceedings of the tenth ACM symposium on access control models and technologies, SACMAT '05, pp 147–158, New York, NY, USA. ACM
65. Freedman MJ, Morris R (2002) Tarzan: a peer-to-peer anonymizing network layer. In: Proceedings of the 9th ACM conference on computer and communications security, CCS '02, pp 193–206, New York, NY, USA. ACM
66. Rennhard M, Plattner B (2002) Introducing MorphMix: peer-to-peer based anonymous internet usage with collusion detection. In: De Capitani di Vimercati S, Samarati P (eds) Proceeding of the ACM workshop on privacy in the electronic society (WPES-02), New York. ACM Press, pp 91–102
67. Choffnes DR, Duch J, Malmgren D, Guierma R, Bustamante FE, Amaral L (2009) Swarmscreen: privacy through plausible deniability in P2P systems. Technical report, Northwestern EECS Technical Report
68. O'Donnell CW, Vaikuntanathan V (2004) Information leak in the chord lookup protocol. In: Proceedings of the fourth international conference on peer-to-peer computing, P2P '04, Washington, DC, USA. IEEE Computer Society, pp 28–35
69. Borisov N, Waddle J (2005) Anonymity in structured peer-to-peer networks. Technical Report UCB/CSD-05-1390, EECS Department, University of California, Berkeley
70. Mislove A, Oberoi G, Post A, Reis C, Druschel P, Wallach DS (2004) AP3: cooperative, decentralized anonymous communication. In: Proceedings of the 11th workshop on ACM SIGOPS European workshop, EW 11, New York, NY, USA. ACM
71. Clarke I, Sandberg O, Wiley B, Hong TW (2001) Freenet: a distributed anonymous information storage and retrieval system. In: international workshop on designing privacy enhancing technologies: design issues in anonymity and unobservability, New York, NY, USA. Springer, New York, Inc, pp 46–66
72. Dierks T, Rescorla E (2008) The transport layer security (TLS) protocol version 1.2. RFC 5246 (Proposed Standard)
73. Rescorla E, Modadugu N (2006) Datagram transport layer security. RFC 4347 (Proposed Standard)
74. Kent S, Seo K (2005) Security architecture for the internet protocol. RFC 4301 (Proposed Standard)
75. Adya A, Bolosky WJ, Castro M, Cermak G, Chaiken R, Douceur JR, Howell J, Lorch JR, Theimer M, Wattenhofer RP (2002) FARSITE: federated, available, and reliable storage for an incompletely trusted environment. In: Proceedings of the 5th symposium on operating systems design and implementation, OSDI '02, New York, NY, USA. ACM, pp 1–14
76. Information Technology Laboratory, NIST, Gaithersburg, USA. In: FIPS 197. Advanced Encryption Standard (AES)
77. Bryan DA, Lowekamp B (2006) Innovations in peer-to-peer communications. In: Proceedings of the 2006 Virginia Space Grant consortium research conference

78. Cao F, Bryan DA, Lowekamp BB (2006) Providing secure services in peer-to-peer communications networks with central security servers. In: AICT-ICIW '06: Proceedings of the advanced international conference on telecommunications and int'l conference on internet and web applications and services, p 105, Washington, DC, USA, 2006. IEEE Computer Society
79. Qureshi A, Rifa-Pous H, Megias D (2013) A survey on security, privacy and anonymity in legal distribution of copyrighted multimedia content over peer- to-peer networks. Technical report, IN3-Universitat Oberta de Catalunya
80. Brands SA (2000) Rethinking public key infrastructures and digital certificates: building in privacy. MIT Press
81. Touceda DS, Camara JMS, Villalba LJG, Marquez JT (2011) Advantages of identity certificate segregation in P2PSIP systems. IET Commun 5(6):879–889
82. Lu Y, Wang W, Bhargava B, Xu D (2006) Trust-based privacy preservation for peer-to-peer data sharing. IEEE Trans Syst Man Cybern Part A: Syst Hum 36(3):498–502
83. Dingledine R, Mathewson N, Syverson P (2004) TOR: the second-generation onion router. In: Proceedings of the 13th conference on USENIX security symposium, vol 13. SSYM'04, Berkeley, CA, USA. USENIX Association, pp 21–21
84. Wallach DS (2003) A survey of peer-to-peer security issues. In: Proceedings of the 2003 Mext-NSF-JSPS international conference on Software security: theories and systems, ISSS'03. Springer, Heidelberg, Germany, pages 42–57
85. Gheorghe G, Lo Cigno R, Montresor A (2010) Security and privacy issues in p2p streaming systems: a survey. In: Peer-to-peer network and applications