Distributed Social Platforms for Confidentiality and Resilience

Enrico Franchi, Michele Tomaiuolo
University of Parma, Italy

ABSTRACT
Social networking sites have deeply changed the perception of the web in the last years. Although the current approach to build social networking systems is to create huge centralized systems owned by a single company, such strategy has many drawbacks, e.g., lack of privacy, lack of anonymity, risks of censorship and operating costs. These issues contrast with some of the main requirements of information systems, including: (i) confidentiality, i.e., the interactions between a user and the system must remain private unless explicitly public; (ii) integrity; (iii) accountability; (iv) availability; (v) identity and anonymity. Moreover, social networking platforms are vulnerable to many kind of attacks: (i) masquerading, which occurs when a user disguises his identity and pretends to be another user; (ii) unauthorized access; (iii) denial of service; (iv) repudiation, which occurs when a user participates in an activity and later claims he did not; (v) eavesdropping; (vi) alteration of data; (vii) copy and replay attacks; and, in general, (viii) attacks making use of social engineering techniques. In order to overcome both the intrinsic defects of centralized systems and the general vulnerabilities of social networking platforms, many different approaches have been proposed, both as federated (i.e., consisting of multiple entities cooperating to provide the service, but usually distinct from users) or peer-to-peer systems (with users directly cooperating to provide the service); in this work we reviewed the most interesting ones. Eventually, we present our own approach to create a solid distributed social networking platform consisting in a novel peer-to-peer system that leverages existing, widespread and stable technologies such as distributed hash tables and BitTorrent. The topics we are mainly concerned with are: (i) anonymity and resilience to censorship; (ii) authenticatable contents; (iii) semantic interoperability using activity streams and weak semantic data formats for contacts and profiles; and (iv) data availability.

INTRODUCTION
Nowadays, millions of people of any age and gender regularly access Online Social Networks (OSNs) and spend most of their online time social networking. According to Boyd and Ellison (2008), teenagers have a clear understanding of privacy related issues; however, the same does not apply to some adults that (i) did not even use email and other basic Internet services before the social networking revolution (Stroud, 2008) and (ii) not only have limited computer-related technical skills, but they also lack risk consciousness about privacy issues. Moreover, many people are becoming uncomfortable with the presence of their employers in the same social networking systems, because some personal data may leak in their corporate environment due to privacy configuration errors (Skeels & Grudin, 2009).
However, privacy threats can also come from the service providers. In fact, even if the social networking systems are greatly dissimilar in their user base and functionality, they are almost always centralized systems. Because of their centralized nature, a simple browser-based user experience is possible and, moreover, many algorithms, e.g., friend suggestion, are far easier and more efficient to implement. A minor drawback is that scaling centralized systems to tens or hundreds of million of users is not an easy task. At any rate, while we consider this drawback as a minor one from a technical point of view, since the problem can be solved providing enough resources, it becomes a huge social drawback, because for
most companies advertisement is the main source of income and, consequently, they have strong motive to make it as precise as possible, typically mining user provided data. This behavior poses serious threats to privacy and data protection issues. In fact, many social networking systems have very demanding terms of service, essentially asking their users a non-exclusive, transferable, sub-licensable, royalty-free, worldwide license to use content that they submit (Facebook, 2011; Twitter, 2011). Moreover, social networking sites guide their users into “walled gardens”, without giving users full control over their own information because such information constitutes much of their company value (Shankland, 2011; Berners-Lee, 2010).

A second feature of centralized systems is that service providers are in the position to effectively perform a-priori or a-posteriori censorship, or to disclose all the information they have, no matter how private, to other entities. They can perform such actions either motivated by selfish interests or forced under legal terms and other forms of pressure. Considering that: (i) no single centralized entity can withstand the operative costs of a large scale social networking system without a solid business-plan; (ii) most business plans are based on targeted advertisement; and (iii) even if a service provider would be fair with its user's data, it would remain vulnerable to legal requests to disclose such data, we favor a P2P approach.

In the first place, P2P systems essentially achieve automatic resource scalability, in the sense that the availability of resources is proportional to the number of users. This property is especially desirable for media sharing social networking systems, considering the exceptionally high amount of resources needed. Moreover, regarding censorship issues, a P2P system essentially solves them by design. Without a central entity, nobody is in the position of censoring data systematically, nor may be held legally responsible for the diffusion of censorable data: the sole owners and responsible of the data are the users themselves. However, P2P systems, and especially those based on a Distributed Hash Table (DHT), may be liable to attacks meant to disrupt the system functionality (Urdaneta et al., 2011); in this particular scenario the severity of such attacks may be mitigated using the social network itself as source of human trust relationships, which make Sybil attacks harder to succeed (Lesniewski-Laas, 2008; Yu et al., 2006).

Privacy, on the other hand, is typically solved using key systems and cryptography.

In the rest of this work, first we introduce some background information regarding: (i) how P2P systems have been successfully used to provide censorship-resistant systems, although without social focus; (ii) how DHTs work, which are their main weaknesses and how they can be overcome; and (iii) how the apparently contrasting requirements of social networking systems (privacy and information sharing) can be reconciled in a distributed system, using modern cryptographic techniques. Then, we introduce the security requirements of a social platform and the main attacks it is susceptible to. Eventually, we present the most widely known distributed social platforms, both in the general context of social networking and in the more reduced context of micro-blogging; then we discuss Blogracy, a distributed social networking platform we have built with features relevant to micro-blogging, such as: (i) anonymity and resilience to censorship; (ii) authenticatable contents; (iii) semantic interoperability using activity streams and weak semantic data formats for contacts and profiles; and (iv) data availability.

BACKGROUND

The P2P community has a long and successful history of solutions to issues regarding various aspects of system and information security, in particular anonymity and privacy. Part of the attentions that have been devoted to the subjects is relative to the intrinsic characteristics of the P2P medium, especially when used for file sharing, where often published data is public by default and consequently security by obscurity is not even a plausible illusion. Moreover, although the main focus of P2P systems was and is information sharing, early legal disputes, involving popular services like Napster, made clear the importance of privacy and anonymity. Eventually, the attacks that P2P systems had to withstand were not only of legal
nature, but also directed against their object storage and search infrastructures (Biddle et al., 2002), and consequently security became a very important topic in the community. These ideas led to the construction of uncensorable and resilient information sharing systems such as Freenet. The idea was that because of their distributed nature, such systems would be movable targets for traditional attackers. We present the main results in that area in Subsection “Decentralized and anonymous content publishing”.

The most widespread P2P systems are implemented using a Distributed Hash Table (DHT), i.e., a decentralized distributed system providing the interface of a traditional hash table, but where the data is distributed among the participants of the distributed system. DHTs work well in practice, still there are a number of vulnerabilities that have to be considered when basing a system over a DHT. They are discussed in Subsection “Vulnerabilities of Distributed Hash Tables”.

Modern distributed social platforms are strongly influenced by the earlier efforts. However, the duality between information sharing and privacy is even more evident. In fact, a privacy-aware social platform should be able to deliver the information to each of the intended recipients and to none of the other users. Unfortunately, in an open file sharing context, access to the data transiting on the P2P medium cannot usually be restricted, leaving cryptography as the only viable solution. The problem is that the set of recipients needs to be set on a per-datum basis. Access may be limited to the members of a circle of personal acquaintances, or authorized subscribers of a news channel. The problem regards also other distributed information storage and retrieval systems and is similar to the one faced by broadcasters of copyrighted materials, who need to distribute protected content over potentially untrusted channels. In this kind of settings, a broadcaster (e.g., a pay-per-view television) would want to be able to send a ciphered content over a public channel, making it readable only by a dynamically selected subset of all receivers. Nowadays, the more promising solution to this kind of issues is Attribute-Based Encryption, which we present in Subsection “Attribute-Based Encryption”.

**Decentralized and anonymous content publishing**

Clarke et al. (2001; 2002) developed Freenet, which is one of the first and more complete implementation of distributed information storage and retrieval system. It is essentially a cooperative distributed file system. Freenet is intended to pave the way for a non-censorable and secure global information storage system characterized by: (i) privacy and anonymity of information producers and consumers; (ii) data location independence; and (iii) lazy replication. Freenet is currently implemented as an adaptive peer-to-peer network, where each node provides both (i) a local data-store, which is made available to the network for reading and writing, and (ii) a dynamic routing table, which associates some nodes with the keys they are thought to hold, on the basis of some heuristics. The system is able to manage two kinds of keys, that are (i) content-hash keys, calculated as the SHA-1 hash of the file content, or (ii) signed-subspace keys, which are the hash of a namespace, concatenated with the hash of the user’s public key that defines the namespace. A signed-subspace key may be associated directly with a content file, or it may be associated with an index file, listing references to some other files, in the form of content-hash keys. Anonymity is obtained through a mix-net scheme. The routing algorithm operates over different kinds of networks, with both “OpenNet” connections (which may involve any node of the network) and “DarkNet” connections (which have to be defined explicitly and are only known to friend nodes). In either case, the protocol is not reliable, i.e., it operates as a “best effort” system and offers no guarantee that a shared file is eventually found. Moreover, it has not been fully analyzed nor evaluated on a very large network. One of the simplest ways to access Freenet is via FProxy, a software which allows users to browse “freesites”, i.e., web sites that store their contents on Freenet instead of a web server. Various popular forum systems are built on Freenet, including Frost, FMS, FreeTalk, but the solution for filtering the large amount of spam coming from anonymous sources, without opening the door for censorship, is not yet clear.
OsirisSPS (Serverless Portal System) ([http://www.osiris-sps.org/](http://www.osiris-sps.org/)) is a framework for creating community-oriented web sites and forums distributed over a peer-to-peer network. It uses a Kademlia DHT for portal distribution and a reputation mechanism for site administration; unfortunately, its source code has not yet been published. The system is anonymous and prevents any association between a user ID and his real identity or network location. User posts are signed and private messages are encrypted. Content is distributed and replicated over multiple nodes, and can be accessed from participating nodes even when they are off-line. Reputation management is one of the most distinguishing aspects of the program: each user is free to rate other users on the basis of their contribution to the portal, and, consequently, each local node is able to process the pages and remove the negative contents (possibly spam or disturbing messages). Other than so-called Anarchists portals, the system allows Monarchists portals, where some users are appointed as administrators and are the only ones able to rate and control other users’ contributions. To simplify access to the portals, a web gateway, named Isis, has been developed. An Isis system does not store information locally, but instead forwards all requests over the Osiris network. Since anonymity cannot be guaranteed for this kind of access, Isis is limited to read-only operations.

Various anonymization services are available, which can also be integrated into more complex applications as an underlying communication network layer. The technologies vary from relatively simple proxies, which protect users’ communications from unauthorized spilling and decouple their actions in the network from their actual network locations, to mix-net schemes based on multiple relays and various envelops and encryption layers. The latter services are often built as some variation of Chaum’s (1981) mix-net scheme, which inspired also the more famous Onion routing scheme used by Tor ([Goldschlag et al., 1999](http://www.torproject.org/)) and the one used by I2P ([Zzz & Shimmer, 2009](http://www.i2p-project.org/)).

From the early experiences on anonymous sharing of resources, some mechanisms were abstracted and developed along some common guidelines. This is the case of Distributed Hash-Tables (DHT), which has become a common component of many modern distributed systems. DHTs associate values with keys, similarly to regular hash-tables, however, the storage of all key-value pairs is distributed among all nodes of the system. In popular peer-to-peer applications, in particular, the Kademlia DHT is an established choice, considering that it is used in the major file-sharing platforms, including eMule and BitTorrent.

Another option is Pastry, which is a realization of the same DHT general concept. In Pastry, however, the key space is assumed circular. Pastry has become the basis of a number of content sharing platforms, thanks to the early work which lead to the realization of PAST, a distributed file system with automatic replication and reliable routing. Castro et al. (2002) built Scribe, an application level multicast infrastructure on top of the Pastry DHT, which was used in a number of projects for peer-to-peer collaboration and dissemination of information. Scribe creates and manages multicast groups on top of Pastry. Any Scribe node can create a group, providing a group ID and some credentials to be used for access control. Other nodes can then join the group or send multicast messages, which are delivered to all members. Multicast messages are delivered by some forwarder nodes, which form a multicast tree. Forwarder nodes themselves are not required to be part of the group, instead they automatically become forwarders if they are on the Pastry route of some new member of the group, when it sends a join request. The delivery mechanism is described as best-effort, without strong guaranties about actual delivery and arrival order of messages. However, some extensions are suggested to obtain stronger reliability, including automatic fault detection and hailing of the multicast tree, sequential numbering of messages and replication of information sources. Simulation results suggest the basic mechanism achieves acceptable delay and link stress when compared to IP multicast, with good load balancing among the nodes of the network.
Vulnerabilities of Distributed Hash Tables

Although DHT-based systems overcome the weakness of a single point of failure, there are some well-known and important vulnerabilities in DHT-based systems:

- **Sybil attacks**, or node insertion attacks, where multiple nodes are created in the network, each of them representing fictitious identities but all belonging to a single user
- **routing attacks**, which collectively use Sybil nodes to inject ad-hoc entries in the routing table of other nodes and thus disrupt the correct message routing procedure
- **eclipse attacks**, which use routing attacks to partition the network in distinct connected components, with the ultimate goal of separating a set of victim nodes from the rest of the overlay network
- **storage attacks**, where Sybil nodes are used to provide bogus responses to queries
- **publish attacks**, based on index poisoning, where essentially some bogus content is deliberately spread to the index nodes responsible for other files or keywords; publish and Sybil attacks are in a sense orthogonal; in fact, publish attacks can exploit Sybil nodes, if available, but can even misguide good nodes to spread false information

Many of the proposed countermeasures to securing P2P networks are based on some notion of “trust” among peers. Depending on the approach they use to evaluate and manage trust relationships among peers, those countermeasures can be divided in two main groups: (i) credential and policy based, (ii) reputation based. Each of them has some context where it can be applied more successfully, though introducing additional complexity and other aspects of security concern. Relying on a Certification Authority means providing it with the power to issue and retract certificates, thus possibly open the way to masquerading attacks, like in the recent cases involving Comodo and Diginota (Pranata et al., 2011), or even detain or generate all the current private keys of the platform principals, which in turn means accessing all protected resources. Reputation systems, on the other hand, need to carefully balance the necessity to exclude rogue nodes from the platform operations, with the risks related to bogus feedbacks and collusions to confuse or subvert the evaluation of peers’ reputation. Cheng & Friedman (2005), for example, propose a general asymmetric reputation algorithm, resistant to Sybil nodes.

Another approach that is possible, is to rely on the structural redundancy of DHT systems, to exclude rogue nodes. Some solutions are proposed, as an application of consensus algorithms to peer-to-peer networks, including the classical Byzantine agreement. This solution is proved to work if less than 1/3 of the n nodes are “traitors” (n > 3t, where t is the number of traitors). Thus, the Byzantine protocol alone may not succeed in blocking attackers, if those are allowed to create an arbitrary number of Sybil nodes. Moreover, the complexity of the agreement protocol, where the number of messages depends exponentially on the network cardinality (Tanenbaum, 2006), make it completely inapplicable on the scale of a global peer-to-peer network, but only to sub-networks.

Along with other similar research works, Lesniewski-Laas (2008) and Yu et al. (2006) propose to exploit the users' social graph to increase resistance against Sybil attacks. Viswanath et al. (2010) analyze various protocols of this type and compare them with previous general community detection algorithms, which all search for a cluster of interconnected nodes around a trusted node. Interestingly, they suggest that networks with such a community structure could be particularly vulnerable to specialized Sybil attacks. Concluding, while being subject to a number of possible attacks, some DHT systems have proved robust enough to continue their operation, mainly thanks to their intrinsic redundancy. In Kademlia implementations, for example, the stable nodes tend to remain longer in the routing tables, thus exhibiting some resistance to the malicious behavior of new rogue nodes introduced into the network. Also, the severity of Sybil attacks can be reduced if nodes are not able to arbitrarily choose their own identifiers, and if the number of nodes running in the same local network is constrained. Cholez, Chrisment & Festor (2010) propose to compare the theoretical distribution of node IDs after a lookup process, demonstrated to be geometrical, with real node IDs, to detect large Sybil attacks. Urdaneta et al. (2011) provide a
detailed analysis of threats to DHTs, together with some proposed countermeasures. While underlining the existing vulnerability to Sybil attacks, authors conclude that “Current DHT deployments are not specifically designed to tolerate the presence of malicious nodes. However, most of them are based on Kademlia, which provides relative security by using data replication and a redundant routing mechanism similar to wide paths.”

Attribute-Based Encryption

In social networking and micro-blogging applications it is often desirable to make some content available to a restricted audience, only. Access may be limited to the members of a circle of personal acquaintances, or authorized subscribers of a news channel. Exploiting traditional public key cryptography and multicast group key management, it is possible to deliver a secret session key to intended recipients of confidential messages. This requires to rekey users periodically, with a computational complexity and message overhead which is linear with N, the cardinality of the group. As an alternative solution, group members can be organized in a multicast tree, reducing the cost of rekeying to log(N) (Canetti, 1999).

Another recently emerging approach is to publish content, possibly on an insecure medium, which can be decrypted only by users with proper attributes, as required by the content publisher's policy. These Attribute-Based Encryption protocols are effectively an extension of the Identity-Based Encryption (IBE) protocol, proposed as a theoretical concept by Shamir (1984). The first practical implementation was described by Boneh & Franklin (2001). In an IBE system, any unique identifier (e.g., e-mail address or OpenId url) can function as a public key. This possibility largely reduces the need for public key infrastructure to distribute certificates. In fact, in IBE schemes users can calculate the public key corresponding to any other identifier they know, after acquiring at startup just a single set of public configuration parameters. To decrypt the received message, instead, the recipient has to download his own private key from a central authority, the Private-Key Generator (PKG). This operation is usually done only once, so that the central authority is not required to be constantly online. This contrasts with traditional public-key infrastructures, in which private keys are chosen randomly, and their corresponding public keys have to be certified. On the other hand, a PKG is more sensitive than a traditional Certificate Authority (CA) in that, if the master private key is compromised, then it can be used to generate the actual private keys of all users. In fact, key escrow is an inherent property of IBE systems. A traditional Certification Authority, if compromised or obliged to, at worst can forge new certificates and certificate revocation information; i.e., it will make future usage of the system insecure, but it will not necessarily be able to read all messages encrypted with previously assigned keys.

After the creation of a working IBE system, advances were made quite rapidly in the field. Sahai and Waters (2005) introduced the concept of Fuzzy-IBE, providing IBE schemes with an error-tolerance which proves handy for encryption using biometrics. Sahai and Waters also introduced Attribute-Based Encryption (ABE). In an ABE system, both the user’ private credential and the cipher-text are associated with a set of attributes. A user can decrypt the cipher-text only if the attributes of his private credential match those of the cipher-text. Based on ABE, Goyal et al. (2006) proposed a key-policy attribute-based encryption (KP-ABE) scheme. Soon after, Bethencourt et al. (2007) constructed the first cipher-text-policy attribute-based encryption (CP-ABE). In CP-ABE, the user is provided with a private credential which attests a set of attributes, while an access policy is embedded directly into cipher-texts. The attributes associated with a user's private credential need to satisfy the access policy of the cipher-text, for the user being able to decrypt the cipher-text. The definition of KP-ABE systems is reversed with respect to CP-ABE systems. In KP-ABE, an access policy is embedded into a user private credential, while cipher-texts are associated with a set of attributes. The attributes of the cipher-text need to satisfy the access policy of the user, for the user being able to decrypt the cipher-text. In both schemes, the required overlapping of attributes can be defined as a k-of-n function, i.e., at least k attributes out of n in the policy
have to be matched. For instance, the common AND or OR operators can be defined as functions with $k=n$ or $k=1$, respectively. Moreover, both KP-ABE and CP-ABE allow users to delegate a subset of their own authorized access permissions, or attributes, to other users, thus facilitating the management of group membership, particularly in those cases where a unique group manager cannot be supposed to be continuously available or being able to manage a large group membership.

SECURITY IN DISTRIBUTED SOCIAL NETWORKING SYSTEMS

In the previous Section, we gave some context regarding P2P systems, their main low-level security issues and typical solutions. Both the issues and the countermeasures are rather technical. On the other hand, we have an intuitive understanding of the main security requirements of a generic information system, which can also be extended on a social networking system. Nonetheless, in this Section we briefly describe such requirements to avoid ambiguity. Eventually, in the second part of this Section, we focus on the typical vulnerabilities of a social networking platform, with special regards for those affecting a distributed one (e.g., in a distributed setting everything occurs on an insecure medium, while in the typical business environment most operations are performed on an intranet, that has higher security guarantees with regards to eavesdropping).

Security requirements of OSNs

The users of information systems have various types of security requirements, including: confidentiality, integrity, accountability, availability and anonymity. The same security requirements can be applied to social networking platforms as well. In this Subsection, we provide a short overview of these security requirements and how they apply to the context of online social networks.

Confidentiality. Any private datum, stored by service providers or communicated to other users by the means offered by the social networking platform, should remain confidential. Related information (such as when the communication occurred or when a specific datum was stored or even when any datum was stored) should remain confidential as well, since eavesdroppers may be able to infer information about a user's activities not only from the content of the messages exchanged, but also from the pattern of message flow on the system. Users may also want to keep their physical and network location confidential. Users should be allowed to decide if their presence (including fine-grained profile data, attributes, contacts) will be available through public directories. Platform security policies, on the other hand, may apply different rules to users who chose to be completely anonymous or operating under a pseudonym, possibly reversible by some privileged authority. Audit logs are important for accountability. But since they list all of a user's important activities on the platform, their content must remain confidential and accessible only to authorized administrators.

Integrity. Social platforms must protect users from unauthorized modification of their profile and messages. The integrity of shared resources and of the whole infrastructure must also be protected from unauthorized modifications. Intentional attacks against a user's communications can be made by changing the content, source or destination of a message, replacing or deleting an entire message, replaying an old message. The integrity of user communications in social platforms also relies on the features of lower-level protocols (e.g., TLS or plain sockets).

Accountability. Each user should be authenticated and audited, maintaining a log of all relevant events, in order to hold him accountable for his own actions. Each log record should at least include the name of the user responsible for the event, time, type and result of the event. The rules applying to the management of audit logs greatly vary in different legal and social contexts. Audit logs must also be protected from lower level failures. Accountability is important for protecting the resources of the system and also for building trust among users. A formally correct user, according to security policies, could still intentionally attempt at deception or spread false information, *bona fide*. Additional auditing may be helpful, especially if reputation is valued in the community. In communities where reputation is held into
account, the social networking infrastructure should protect it from various threats, including masquerading attacks.

**Availability.** Especially in the case of professional communities, the social networking infrastructure should ensure high availability of both data and services. The social platform should be able to handle the requests of scores of visitors and regular users, or risk suffering common denial of service threats. Shared resources should be available and allocated according to a fair policy, or at least to a graceful degradation of service quality. The infrastructure should be able to detect and recover from various failures. In some cases, users may be directly involved in the recovery process. Ensuring confidentiality, integrity and accountability requires additional computational resources, disk space and network capacity, thus influencing the system availability.

**Identity and anonymity.** In social networking platforms, accountability of users always needs to be balanced with privacy requirements. In some cases, users may be asked to provide their real world identities, while in other cases only their virtual identities are relevant. Especially in the first case, the platform should provide means to keep profile information secret from other users, while still maintaining a form of reversible anonymity, if necessary for legal or other accountability reasons. The policy of collection and use of audit information must be available to users in a comprehensible form. In human communities, anonymity is important in allowing the adoption of unpopular viewpoints, lifestyles and behaviours. On the other hand, anonymity makes the development of trust among users difficult or impossible.

**Security threats to OSNs**

Social networking platforms are susceptible to different types of attacks. For better analyzing these attacks, it is useful to identify the main abstract components of a generic social networking platform, corresponding to different functional aspects of those systems. We identify four main components:

- **The social networking** component, which manages and protects access to the users' personal profiles and the social relationships among users.
- **The content management** component, which manages and protects access to all user generated content, including personal status updates, comments, links to other content, photos and multimedia galleries.
- **The infrastructure services** component, which provides the basic infrastructure services needed to run the social networking platform, including storage and replication services for content and profiles, information indexing and routing, management of users' online presence.
- **The communication and transport** component, encapsulating basic inter-networking and ad-hoc networking functionalities.

Attacks can be directed to each of the different layers we mentioned; some attacks may target more than one layer or there may be attack variants targeting different levels, but with roughly the same logic. Moreover, we can differentiate between two different kind of attackers: (i) **intruders**, i.e., users accessing the system without proper authorization, or (ii) **insiders**, i.e., regular users or entities participating in the systems operations, assuming malicious behaviors. From the users point of view, malicious behavior can also be attributed to the service provider. In the rest of this Subsection we review the main kinds of attack.

**Masquerading.** When a rogue user disguises his identity and claims the identity of another user, the former is said to be masquerading. Masquerading may be attempted by an attacker either during a conversation or while registering his own profile, for deceiving other users or the whole social networking platform. Simple impersonation, by cloning the victim's profile from the same platform or by porting profile data from a different platform, may easily lead the attacker to gain trust from the victim's contacts. Especially in communities where reputation is valued, masquerading can damage the user whose identity has been stolen, as it can damage other users eventually deceived. Sometimes, masquerading is the first step to gain access to infrastructure services and resources to which the attacker is not entitled.
Another possibility is that the attacker is pretending to be another user in order to shift the blame for any liable action. A particular type of masquerading occurs when a rogue platform pretends to be a legitimate social networking platform, misleading unsuspecting users, with the goal to acquire sensible information (phishing) or engaging in other harmful activities.

**Unauthorized Access.** Users who have not been granted adequate permissions for accessing some services and resources, may attempt to circumvent the security mechanisms and policies of the system and gain unauthorized access. In a social networking platform, any user who has access to some profiles and messages can harm their legitimate owners. Accessing data without proper authorization allows also an untrusted user to produce such harm. The collection of existing data is the basis of profiling attacks. These data may also supply some knowledge for secondary data collection from a wide range of different sources, including other OSNs. Remote access can also occur at system level. In this case the attacker may directly gain control of all resources.

**Denial of Service (DoS).** The services and communications at the infrastructure level can be disrupted by common denial of service attacks. Social networking platforms are also susceptible to all the conventional denial of service attacks aimed at the underlying operating system or communication protocols. In addition to attacking the whole infrastructure of a social networking platform, users can also launch denial of service attacks against specific users, especially in a distributed platform. For example, repeatedly sending messages or other spam may place undue burden on the recipient users and their systems. Malicious users can also intentionally distribute false or useless information to prevent other users from completing their social activities.

**Repudiation.** In general, repudiation occurs when a user, after having performed some action, later denies that action having happened (at least under his responsibility). Repudiation can be intentional or even accidental. It can also be the result of a misunderstanding, when users have a different view of events. In any case it can generate important disputes. In a sense, nothing can prevent a user from repudiating one of his actions. But a social networking platform can eventually help resolving disputes by providing needed evidence, if it maintains a sufficiently detailed log of events. For users who value their reputation, the availability of such evidence may constitute a valid deterrent.

**Eavesdropping.** The attempt to observe the flow and possibly the content of confidential messages is one of the most classical security threats. Apart from reading the content of messages, which may require cryptanalysis, an eavesdropper may gather useful information by simply observing the pattern of messages and their recipients, for example inferring the type of services being requested. To eavesdrop on other users, an attacker may also exploit the infrastructure and communication services of the platform, e.g. through unauthorized access.

**Alteration.** When a user signs up a social networking service, he starts exposing his profile and content to the platform. An attacker may tamper with the profile and content data published by the victim, with all the messages he communicates to other users and all data used on the infrastructure services. Alteration can also be conducted by the service operator, which provides the facilities for online social networking and may take control of published data. Alteration may take the particular form of filtering, or censorship, when applied systematically for removing undesired content from the OSN.

**Copy and Replay.** Each action in a social network may be subject to copy and reply. In this type of security threat, an attacker attempts to intercept some data and clone it, for retransmitting it later. The interceptor may successfully copy and replay a message, a complete profile or any other data. If those data are not associated with a signature and a timestamp, the repeated reception of such copies may pass unnoticed and accepted as a legitimate action.

**Social engineering.** In a social networking application, a common attack is to psychologically manipulate a user into performing misguided actions. It is similar to a confidence trick or a traditional fraud, but by means of computer-based communications and online social networking, typically to gain access to confidential information. In most cases the victim and the attacker never acknowledged each other directly in real life. In its essence, “social engineering” is associated with social sciences and is defined in
general as an act of psychological manipulation. But recently its usage is becoming increasingly important among computer professionals.

DISTRIBUTED SOCIAL NETWORKING SYSTEMS

In the Introduction, we argued that many of the problems relative to social networking platforms are related to their centralized nature. We think that a viable solution is to distribute the platform, so that no single entity: (i) has to withstand the costs; (ii) is responsible for the data (both legally and operatively); and (iii) owns the whole system. The distribution can occur in two radically different ways:

1. **Federation.** In a federated system, multiple entities cooperate to provide the service. Each of them provides access to the whole system to a subset of its users. The system is perceived as a whole because each of the federated providers keeps information synchronized with the other providers. The users are free to choose a provider they trust (both from a technical reason and from legal/moral ones). Notice that many existing systems already work this way: e.g., emails are a federated service, where multiple servers cooperate. However, a social networking platform has a much higher interactivity with respect to emails. Notice that in federated systems it is still possible that a user is also a service provider (for himself). For instance, it was not uncommon among UNIX users to have their own SMTP server. Nowadays, spam and the consequent strategies such as white/gray-listings made this approach less popular.

2. **P2P.** In a P2P system, *every* user is also a service provider. The whole system is built around this idea, which can be interpreted as a limit case of federated system. The traditional distinctions between servers and clients are blurred, every node both provides services to and requests them from the other nodes. We favor this approach because it maps well the very structure of social networks, that are made of interacting nodes.

In the rest of this Section, we review some of the most popular distributed social networking systems, both federated and P2P. Eventually, we present Blogracy, a P2P system we built to overcome the issues of confidentiality and resilience in social networking systems.

**Federated social networking systems**

Among the federated social networking systems, one of the best known is **Diaspora** ([http://joindiaspora.com/](http://joindiaspora.com/)). Diaspora is being implemented in Ruby and released as open source. Users can participate in the network by setting up their own server, which is named a “pod”, and can host the content produced by various users. Otherwise they can exploit already existing pods which can host their content and their social connections. User relations and information flows are asymmetrical, i.e., a user’s content is only distributed and disclosed to authorized followers. Diaspora servers communicate by means of an ad-hoc federation protocol and the standard Salmon protocol ([http://www.salmon-protocol.org/](http://www.salmon-protocol.org/)) for comments. They exchange semantically annotated messages in various situations, including: discovery of information about hosted users, notification of acceptance for sharing information, publication and possible retraction of posts, publication and possible retraction of comments and “like” flags (either from the user or from others) on one of the user’s post, private conversations and messages, profile information. The security model being proposed for implementation is still under discussion, but would probably include encryption of a single post trough a session key. The session key has to be sent individually to the audience of the post; thus, it needs to be encrypted repeatedly, using the public key of each intended recipient.

Another attempt in a similar direction is **StatusNet** ([http://status.net/](http://status.net/)), formerly known as Laconica. StatusNet is an open source project providing similar functionalities to those found on Twitter. It is implemented in PHP and adheres to the OStatus standard protocol for the interconnection of various servers. The first deployment was the identi.ca micro-blogging service. Among the most interesting features, StatusNet shows quite strong interoperability with other networks:
• ability to send updates via (i) the XMPP protocol, (ii) SMS, (iii) the Salmon protocol;
• support for the OpenID authentication;
• compatibility with Twitter at the API level;
• integration, in various ways, with both Twitter and Facebook.

Apart from automatic management of URL-shortening, StatusNet also handles various semantic contents associated with posts, including: (i) geolocation and maps; (ii) attachments and links to external resources; (iii) both hash-tags and bang-tags for groups. From the security point of view, StatusNet essentially relies on a profile URI, which needs to be based on a secure transport for assuring authenticity and confidentiality of exchanged data. The OStatus and Salmon protocols allow additional levels of security, but still require a secure transport at handshake.

Peer-to-peer social networking systems

PeerSoN (Buchegger & Datta 2009; Buchegger et al. 2009) is a system designed to provide encryption, decentralization and direct data exchange in the field of social networks, dealing with privacy and connectivity issues. The implementation is based on a two-tier architecture. The first tier is based on a Distributed Hash Table (DHT) and handles the look-up functionalities; basically it allows to find users and data over the social network. Unfortunately, the originally chosen DHT implementation, OpenDHT, later became overloaded and unusable and had to be switched off (Buchegger et al., 2009). As a consequence the PeesSoN prototype is presently run as a centralized service, for testing and evaluation. With regards to security and encryption, the first prototype of PeerSoN is designed around a Public Key Infrastructure (PKI). Given the centralized or hierarchical nature of PKI, this solution hardly matches the needs of a decentralized network. In fact, Buchegger et al. (2009) hint at some studies being conducted for removing a centralized PKI and for developing a more efficient approach, but the result is not yet clear, as the initial focus has not been oriented on the encryption and privacy issues, but to make online social networks distributed.

A more interesting part of the system design is the description of prototype protocols. In PeerSoN each user has a unique ID, possibly computed as a hash of the user’s email. In the DHT, various values are associated with this ID, representing the various locations or machines used by the user to connect to the network, and the status of each location, connected or not. Then, to check for updates from a particular user, the filename of his index file is used as a key in the DHT, and corresponding locations possessing the file are returned, associated with the version number of the stored index file. It is worth noticing that this protocol is a bit different from common file sharing, where a hash of the file content, and not its name, is used as its DHT key. The index file contains a list of newly generated content, which has to be searched for and downloaded in a similar way. Another possibility, which is considered by the PeerSoN system, is the direct pushing of new content to supposedly interested peers. Some measures are suggested to avoid receiving spam and other undesired or malicious content, especially on constrained devices and connections. In particular, gray and black lists are cited, but without providing any detail about adopted reputation mechanisms.

Safebook (Cutillo et al., 2009) is based on a DHT and a network of socially close peers, defined Matryoshka. Peers in a user's Matryoshka are trusted and support the user by anonymizing communications and replicating content and profile information. Safebook exploits a more traditional certification authority. In fact, a user's public key cannot be calculated from his identity, and all public/private key pairs are generated locally by the peers.

LifeSocial (Graffi et al., 2010) is a prototype developed over FreePastry for DHT indexing and PAST for data replication. It is composed of various mandatory modules, for managing profile, friends, groups and photos. Additional modules are available for chat and whiteboard functionalities.

Blogracy is our own fully distributed P2P micro-blogging platform, based on Bittorrent. Its main focus is: (i) anonymity and resilience to censorship; (ii) authenticatable contents; (iii) semantic interoperability
using activity streams and weak semantic data formats for contacts and profiles; and (iv) data availability. More details are given in the “Design and Features of Blogracy” Subsection of the present chapter.

**Distributed propagation of feeds and updates**

Xu et al. (2010, 2010b) described **Cuckoo** as a decentralized and socio-aware online micro-blogging service. It follows a hybrid approach consisting of: (i) a structured overlay network, Pastry, and a gossip protocol for disseminating micro-news among users with the same interests; and (ii) support for centralized dedicated services, like Twitter, which in fact still store users’ profiles and other data. The peer-to-peer infrastructure is mainly motivated by the goals of reducing bandwidth and storage space on the server side. On this basis, Cuckoo nodes maintain information about social relations, i.e., friends, neighbors, followers and followees. Friend nodes help each other to balance load, thus creating a sort of virtual node. Notifications are dealt with direct push, in the case of normal users, or with gossip propagation, in the case of celebrities and broadcasters.

**FeedTree** (Sandler et al. 2005) is an RSS feed distribution service based on peer-to-peer subscription mechanisms. RSS (Real Simple Syndication) is an XML-based feed format used by websites to list novel content: its basic functioning require interested subscribers to query a server periodically, according to a typical pull approach, thus significantly increasing the load of popular websites. FeedTree proposes a transition towards pushing RSS items over a peer-to-peer network, distributing the load over the nodes of a group multicast tree. For this purpose, FeedTree exploits Pastry and Scribe. To allow a smoother adoption of the service, various levels of integration with existing software are proposed. The easier one includes a “republishing” engine running close to the web-server, sifting new content for peer-to-peer distribution, and a FeedTree proxy running on the client machine, receiving feeds pushed over the peer-to-peer network and providing them directly to the traditional RSS client.

Perfitt & Englert (2010) describe **Megaphone** as a micro-blogging system, based on an optimized, trustworthy peer-to-peer network. In fact, nodes are enabled to sign and encrypt each piece of content they publish, making it verifiable and confidential for subscribers. The basic distribution mechanism is based on Scribe multicast trees. Thus, a subscriber node has to know in advance the node ID of the posters to follow, or at least it has to be able to generate it. The poster’s node ID corresponds exactly to a Scribe multicast group ID. In Megaphone, the node ID is a hash of its public key, and the couple of public/private keys is generated autonomously by each node. The follower is supposed to obtain the poster’s public key via an out of bound mechanism, like an email or a web page, in the process of discovery and acquaintance with the poster. Each message can be encrypted with a secret session key, which is suggested to be changed on a daily basis. The session key is itself ciphered with the public key of each follower. All of these copies of the session key are then packaged together and distributed as a message over the multicast channel. Since the scalability of such a mechanism has quite clear limitations, an alternative solution, based on delegation, is briefly described in the original work. Thus, to improve its scalability towards large micro-blogging environments while preserving confidentiality, the Megaphone system should be adapted to this kind of delegation, which however requires additional join requests and some different but important interaction of the poster node with each subscriber.

**Other social networking systems**

**SCOPE** (Mani et al. 2010) is a prototype for spontaneous peer-to-peer social networking. The main aim of the system is to create ad-hoc social networks based on proximity, i.e., exploiting wireless network infrastructure and services available in the user’s local place. The system is composed of a number of modules, together providing all needed functionalities. One module is designed for social services, i.e., for the management of user profiles, social contacts and groups. Another module uses a DHT to provide distributed storage. A module based on P2PSIP (Bryan et al. 2006) is responsible for the management of sessions, for real-time applications. Then, specialized modules are dedicated to service advertisement and
discovery (yellow pages) and for other tasks. The network itself is organized in a typical hybrid peer-to-peer architecture, with peers distinguished as super nodes and client nodes. While super nodes provide all the important services of the network, including lookup, routing, session management and DHT-based storage, instead, client nodes are supposed to have highly constrained computational resources and bandwidth, and thus are only able to connect to super nodes and exploit the needed services.

**Persona** (Baden et al., 2009) is designed as a set of social networking services. It uses an interesting Attribute-Based Encryption protocol for protecting access to users’ content. It allows each user to create various groups of “friends”, by assigning proper attribute credentials. Content can then be associated with a publication policy and made available only to a restricted audience.

**Design and Features of Blogracy**

While many authors argue for the distribution and openness of social networking and micro-blogging services, few usable implementations exist, either in the field of federated networks or as fully distributed solutions. However, most of the systems and studies described in the previous Section shall be carefully considered since they provide some theoretical foundations and some viable solutions to particular issues, even when they remain at the level of abstract algorithms, simulations and theoretical analysis. Moreover, all these efforts led to the definition of open formats and protocols which are common ground for the interoperability and distribution of social networking applications. On the basis of existing or proposed solutions, we therefore present a new system, which we named Blogracy ([http://www.blogracy.net/](http://www.blogracy.net/) (Franchi & Tomaiuolo, 2012). In the following paragraphs, we describe its main distinguishing features and how they relate to other existing systems and abstract architectures. Our new system is built incrementally over popular services; in fact, from the very beginning, we chose to leverage existing and widespread networks for file sharing, and providing to those enormous communities specific features for micro-blogging applications and for publishing personal activity streams.

From an architectural point of view, Blogracy is built upon a peer-to-peer file-sharing mechanism and two logically separated DHTs. Users in Blogracy have a profile and a semantically meaningful activity stream, which contains their actions in the system (e.g., add a post, tag a picture, comment a video). One DHT maps the user’s identifier with his activity stream, which also contains a reference to the user’s profile and references to user generated content (e.g., posts, comments). These references are keys of the second DHT which are then resolved to the actual files, delivered using the underlying peer-to-peer file-sharing mechanism. These basic file sharing techniques are well tested and in widespread use. In particular, it is common practice to associate files (or file parts, or chunks) with their hash, and using the hash itself to identify a file, to share or download. A quite similar technique was found also in Freenet, as well as most of the following networks.

For all practical purposes, individual users of large networks have to be associated with numerical identifiers or unique strings, since names used in real life are hardly unique. In a typical Trust Management scheme, a user’s public key is used directly to represent the user, so that all contents produced by the user can be easily verified against his public key, which is also his own main identifier. Alternatively, a cryptographic hash of the public key can be used (Li, 2000), without loss of security, and this is exactly the scheme adopted in Blogracy, the hash function being the same as the one used by the DHT.

However, anonymity is an issue also at the lower network level. In fact, if file locations are expressed as plain network addresses, these can be easily associated with a particular person or entity which can be called for taking legal responsibility of shared materials or expressed ideas, if they are deemed illegal or censored for any reason.

The users’ network addresses, in particular, are typically published in DHT entries associated with shared files. This kind of issues applies to a number of contexts but, since they are essentially related to a lower network level, they are best solved at that very level. In fact, various anonymizing technologies exist,
ranging from simple proxies, to complex networks. The latter includes the famous Tor (Dingledine et al., 2004), a network based on the onion routing protocol, but with centralized management, and I2P (Zzz & Schimmer, 2009), also based on a variation of the mix-net scheme, but completely distributed.

Being mainly a platform for resilient micro-blogging, Blogracy is designed to assure new contents are published and distributed as widely as possible, exploiting the most popular and effective file-sharing infrastructure available today. While contacts are not published by default, actually there’s no restriction about access to posts. However, since the core sharing system is completely agnostic with respect to published content, data can be easily encrypted with any cryptographic algorithm, either symmetric or asymmetric, including attribute-based encryption. In fact, Blogracy supports attribute-based encryption. Similarly to Persona, Blogracy privacy model uses attribute credentials for protecting access to sensible content, creating a sort of very flexible personal circles of contacts, i.e., parametrized roles to be assigned to users for granting a certain set of access rights. The encryption scheme is based on the CP-ABE protocol (Bethencourt et al., 2007). A practical and quite simple alternative, which could be easily added to the system, is to encrypt sensible posts with a secret session key, and then encrypt this session key with the public key of each intended reader. This approach would be similar to Megaphone, and it would scale till medium-sized groups, which would probably cover a significant number of use cases.

Considering individual security threats, Blogracy is designed to show resilience against censorship and centralized control over published data. Its completely distributed architecture and the replication of popular data, typical of file sharing systems, provide also resistance against DoS attacks. Moreover, traditional signatures and timestamps, together with CP-ABE, are used as means to protect against eavesdropping, alteration, copy and replay, repudiation, unauthorized access. Conversely, it opposes limited resistance against masquerading and some form of social engineering, exactly for the absence of any centralized control.

**FUTURE RESEARCH DIRECTIONS**

While a number of distributed social networking systems have been proposed, many underlying mechanisms and also various aspects of the high level architecture are yet object of analysis and research. Many of these systems, for example, use some DHT component for basic indexing. However, the robustness of DHTs, and in particular their resistance to Sybil and pollution attacks, is still being discussed. The approaches to this issue vary (Urdaneta et al., 2011), but in general in distributed social networking systems the presence of a global authentication authority is not always possible or desirable. Distributed mechanisms for reputation and trust may contrast with the requirement for anonymous operation, but are still possible under the assumption of secure pseudonyms (Li, 2000). Social relationships of trust among human users can also be used to make the DHT more robust (Lesniewski-Laas, 2008; Yu et al., 2006).

Open peer-to-peer overlay networks usually guarantee a certain level of anonymity, since usually they do not require any registration or authentication. However, the network addresses involved in the network operations may be traced, and the association with a responsible person may be inferred. Anonymizing infrastructures exist, ranging from simple proxies to global mix-net networks, but their availability, operation cost and resilience to technical and legal threats are not yet adequate for all use cases (Dingledine et al., 2004). Fully distributed, automatically and dynamically reconfiguring schemes would better scale to the needs of global peer-to-peer networks (Zzz & Schimmer, 2009).

On the issue of the confidentiality of personal data, solutions for sharing data only for a limited audience may be adapted from other fields. For example, the problem of broadcasting copyrighted material has some similarities with our problem. However, typical solutions were devised with a smaller number of entities in mind, and suffer from scalability problems, especially in the case of rekeying. New cryptographic schemes may be applied, but they need to be generalized for appliance in open networks. For example, ABE encryption schemes are promising (Bethencourt et al., 2007), but they need to work in
an open, multi-authority scenario (Lewko & Waters, 2011), possibly maintaining the confidentiality of exchanged credentials and policies, like with oblivious commitments (Li & Li, 2006).

In distributed social networking platforms, the responsibility of storing all data is shared among users. For this reason, it is necessary to ensure a minimum level of **data availability** even for users with sparse social relations. Rzadca et al. (2010) discuss how data availability in a P2P social application should be modeled and modeling the systems and they estimate expected performances. Such studies are expected to become more common because they avoid the likelihood to deploy a system that cannot work.

Newer social networking systems should inter-operate with the other existing systems, including social networking sites, micro-blogging platforms and all other existing forms of social information systems. Although some standardization efforts are being conducted – including Portable Contacts, Activity Streams, OpenSocial, FOAF –, (i) their acceptance is not yet adequate, (ii) the standards are not yet stable enough and (iii) their implementations are not yet completely compatible.

Finally, in distributed social platforms, algorithms for **friend discovery** are still an argument of research. This issue is related to the intrinsically distributed nature of those systems, where information about users is not available to a single central entity, but is available only to local contacts. Moreover, the issue is complicated when profile data is kept confidential, and disclosed only after some acknowledgement between the users. In those cases, the algorithms and the studies about Automated Trust Negotiations (Winsborough & Li, 2002) may help to define patterns and strategies for facilitating the creation of trust relationships.

**CONCLUSION**

Although the current approach to build social networking systems is to create huge centralized systems owned by a single company, such strategy has many drawbacks, regarding: (i) privacy and anonymity of users; (ii) risks of a-priori or a-posteriori censorship; (iii) ownership and use-rights of the users data; (iv) interoperability with other systems; and (v) costs of the infrastructure, especially when it has to support media files.

In this chapter, we discussed the main security and functional requirements of a social networking platform, that are: (i) confidentiality, i.e., the interactions between an user and the system must remain private unless explicitly public; (ii) integrity, i.e., unauthorized modifications to profiles and other user generated content must not occur; (iii) accountability, i.e., authorized modifications must be traceable; (iv) availability; (v) identity and anonymity. Such requirements can be violated by various threats and attacks, such as (i) masquerading, which occurs when a user disguises his identity and pretends to be another user; (ii) unauthorized access; (iii) denial of service; (iv) repudiation, which occurs when a user participates in an activity and later claims he did not; (v) eavesdropping; (vi) alteration; (vii) copy and replay; (viii) social engineering.

In order to overcome both the intrinsic defects of centralized systems and the general vulnerabilities of social networking platforms, many different approaches have been proposed, either as federated (i.e., consisting of multiple entities cooperating to provide the service, but usually distinct from users) or peer-to-peer systems (with users directly cooperating to provide the service) and in this work we have reviewed the more popular.

Eventually, we briefly discussed our own approach to create a solid distributed social networking platform consisting in a novel peer-to-peer system that leverages existing, widespread and stable technologies such as DHTs and BitTorrent. Blogracy is a micro-blogging social networking system, and consequently we gave priority to the features more important for micro-blogging, such as: (i) anonymity and resilience to censorship; (ii) authenticatable contents; (iii) semantic interoperability using activity streams and weak semantic data formats for contacts and profiles; and (iv) data availability. Although Blogracy is not yet feature-complete, we have created a working prototype, implementing all core functionalities as a layer over a well-tested distributed file sharing system.
REFERENCES


**ADDITIONAL READINGS**


of Peer-to-Peer systems (pp. 68-72).
KEY TERMS & DEFINITIONS

**Distributed Hash Table (DHT):** a mechanism for distributing data among peers in an overlay network. The network may have responsibility for managing actual data storage or simply indexing. Usually a DHT allows to efficiently search for a key, for retrieving and storing corresponding values on the proper node. In fact, the domain of possible keys is subdivided among the nodes, and each node is responsible for a certain subspace. Typically the complexity of the search is logarithmic against the network cardinality.

**Sybil Node:** a node of a peer-to-peer network belonging to an attacker, i.e. an entity which operates to subvert the network operation. Usually a large number of Sybil nodes are generated, possibly using a limited number of actual computers, to collaborate and alter reputation, routing, indexing and/or any other functionality of the peer-to-peer network.

**Darknet:** in general a hidden or obscure network. It may include some web-like infrastructure and content or any other collaborative information systems. Independently of its mechanisms and goals, its distinguishing feature is to be clandestine, in the sense that it is known and accessible only to the acknowledged members of a certain community.

**Social Engineering:** in information security, social engineering is an attack meant to psychologically manipulate a user into performing misguided actions. Exploiting computer-based communications and online social networking systems, it aims at obtaining confidential information, by means of confidence tricks. Social engineering is similar to frauds attempted in more traditional contexts, but is becoming increasingly important among computer professionals. The victim and the attacker need not to know each other in their real lives, and in fact often they never acknowledged each other directly.

**Pseudonymity:** a mechanism for allowing users to operate under a pseudonym, or a “false name”. It is used by people to avoid disclosing their real name and in particular their legal identity. Pseudonymity has been used long before the Internet, but it has acquired particular importance for users operating in global computer networks. The pseudonymity mechanisms available on the Internet may vary in the degrees of anonymity they provide. In fact, pseudonyms may range from unlinkable identifiers, like those provided by some remailer systems, to easily linkable names, where the association with the particular user (or the legal entity) they represent is publicly available.

**Trust Management:** a system for the symbolic representation, creation and management of social trust. In particular, trust can be used for controlling access to protected resources. A request for accessing a resource is accepted only if accompanied with sufficient credentials, according to a local policy. Trust, quantified as a set of access rights, can be further delegated to other agents, creating trust networks. A chain of credentials may be used to represent the trust flow, from the resource manager to the agent requesting access. Automated Trust Negotiation allows to automate trust building, guiding the disclosure of credentials according to privacy policies and negotiation strategies.

**Social Network:** social structure made of agents (individuals) that are connected by relationships.

**Social Networking System:** a software system that allows users to manipulate a representation of their online social networks and to interact with the other users in the system, especially collaboratively discussing user-produced resources (e.g., posts, pictures).