Peer-to-peer overlays for real-time communication: security issues and solutions

Dhruv Chopra and Henning Schulzrinne
Department of Computer Science
Columbia University
New York, NY 10027, USA
Email: {dc2378, hgs}@cs.columbia.edu

Enrico Marocco
Telecom Italia
Via G. Reiss Romoli, 274
10148 Turin, ITALY
Email: enrico.marocco@telecomitalia.it

Emil Ivov
Louis Pasteur University,
SIP Communicator Project
67412 Strasbourg, FRANCE
Email: emcho@sip-communicator.org

***Abstract*—Peer to peer (P2P) networks offer higher robustness against failure, easier configuration and are generally more economical as compared to their client-server counterparts. This has made it reasonable for resource consuming and typically centralized applications like Voice over IP (VoIP) and, in general, real-time communication to adapt and exploit the benefits of P2P. Such a migration needs to address a new set of P2P specific security problems. We go over some of the known issues found in common P2P networks. We then analyze the relevance of such issues and the applicability of existing solutions when using P2P architectures for real-time communication.**

I. Introduction

Peer to Peer (P2P) overlays have become quite popular with the advent of file-sharing applications such as Napster [1], KaZaa [2] and Bittorrent [3]. After their success in file-sharing and content distribution [4], P2P networks are now also being used for applications such as Voice over IP (VoIP) [5], [6] and television [7], [8]. However most of these systems are not purely P2P and have centralized components like the login server in Skype [9] or moderators and trackers in Bittorrent [10]. Securing pure P2P networks is therefore still a field of very active research [11]. P2P overlays can be broadly classified as structured and unstructured. Unstructured overlays are often relatively simple but search operations in them tend to be inefficient. Structured P2P overlays use distributed hash tables (DHT) to perform directed searches which make lookups more efficient in locating data. Throughout this paper we will mostly focus on DHT-based P2P overlays.

When analyzing the various attacks that are possible on P2P systems, it is important to first understand the motivation of the attackers as well as the resources (i.e. computation power, access to different IP subnets) that they would have at their disposal.

Once the threat has been identified, admission control is the first step towards security [12]. Most solutions rely on the assumption that malicious nodes represent a small fraction all of peers. It is therefore important to restrict their number in the overlay.

Other P2P specific security problems that we present here include attacks on the routing of queries, targeted denial of service attacks and attacks on data integrity.

In this paper, after discussing some of the main security issues and proposed solutions for P2P systems in general, we focus on one particular application – real-time communication. The idea behind P2P real-time communication is using the DHTs employed by file-sharing applications, in order to implement services such as registration, user location lookup, and assistance for NAT and firewall traversal. Even if, from a technical point of view, P2P communication sevices may seem similar to file-sharing, table I shows that some important differences, mostly related to privacy and availability, significantly increase security requirements. We discuss how the above security aspects apply to this application and the solutions that seem appropriate.

The rest of this article is organized as follows. We begin with a general presentation of the P2P context today in section II. In section III, we discuss P2P security attackers. We try to elaborate on their motivation, the resources that would generally be available to them, their victims and the timing of their attacks. In section IV, we discuss admission control problems. In section V, we identify the problem of *where* a node joins in the overlay. In section VI, we describe problems related to identification of malicious nodes and the dissemination of this information. In section VII, we describe the issues of routing and data integrity in P2P networks. In section VIII we discuss how issues and solutions previously presented apply in P2P overlays for real-time communication, and in section IX we conclude the paper and enumerate topics that would require future work.

II. Background

The concept behind popular peer to peer applications used for file-sharing and real-time communication over the Internet is fairly simple: peers willing to use the service cooperate to maintain a distributed database to index file and user locations, and use indexed data to establish direct connections for transfering files and exchanging media. Such a distributed index is usually implemented through a distributed hash table that could be maintained by all peers or by only a subset that reply to specific criteria.

A DHT is a distributed system which implements typical hash table functionalities – such as efficiently storing and retrieving key-value pairs – but which uses the hash function to map keys on network nodes instead of memory slots. Figure 3 shows how keys are distributed in ring-based (e.g. Chord [13],

	File-sharing applications	Real-time communication applications
Distributed database	Shared files locations are indexed in a table distributed among peers; often hundreds or thousands per user	User locations are indexed in a table distributed among peers; rarely more than one per user
Availability	Same files are usually available at multiple locations and failures involving single istances are overcame by abundancy of resources; attacks targeting single files need to be addressed to the distributed index	Users are unique; attacks targeting single users may be addressed both to the distributed index and to the user's device directly
Integrity	Attackers may want to share corrupted files in place of popular content, e.g. to discourage users from acquiring copyrighted material; constitute a threat for the service, but not for the users	Attackers may want to impersonate different users in order to handle calls directed to them; constitute a particular threat for the user as, in case of success, the attacker acquires full control on the victim's personal communications
Confidentiality	Shared files are, by definition, readable by all users; in some cases encryption is used to avoid elements not involved in the service to detect overlay traffic	Communications are usually meant to be private and need to be encrypted; evesdropping may reveal sensitive data and is a serious threat for users

TABLE I
KEY DIFFERENCES BETWEEN PEER-TO-PEER FILE-SHARING AND REAL-TIME COMMUNICATION APPLICATIONS.

Kademlia [14], Pastry [15]) and hypercube-based (e.g. CAN [16]) DHTs.

Since DHTs are designed to work in environments characterized by high churn rates, they are inherently fault tolerant and much less vulnerable to denial of service attacks than centralized solutions. However, some of the most common security issues, mainly those regarding integrity and confidentiality, are traditionally addressed on the endpoint and thus the solutions are almost the same in both distributed and centralized environments. Finally, because users are directly involved in the provision of the service and due to the complete lack of control, P2P networks may be subject to a number of attacks caused by malicious peers inside the overlay which traditional systems may usually ignore. In the rest of this article we will address some of the most important issues affecting P2P systems in particular, and will analyze the applicability of existing solutions.

III. THE ATTACKERS

A. Incentive of the attacker

Attacks on networks happen for a variety of reasons such as monetary gain, personal enmity or even for fame in the hacker community. There are quite a few well known cases of denial of service attacks for extortion in the client-server model [17]. One of the salient points of the P2P model is that the services it provides have higher robustness against failure. However, such attacks are still possible against individuals within the overlay if the attackers possess sufficient resources. For instance, a network of worm-affected malicious nodes spread across the Internet and controlled by an attacker (often referred as botnet), could simultaneously bombard lookup queries for a particular key in the DHT. The peer responsible for this key would then come under a lot of load and could crash [18]. However with replication of key-value pairs at multiple locations, such threats can be mitigated.

Attackers may also have other incentives apart from money. With the growth of illegal usage of sharing files with copyrights, record companies have been known to attempt polluting content in the overlays by putting up nodes with corrupt chunks of data but with correct file names to degrade the service [19] and in hope that users would get frustrated and stop using the service. Attacks can also be launched by novice attackers who are there attacking the overlay for fun or fame in a community. These are perhaps less likely to be successful or cause damage, since their resources tend to be relatively limited.

B. Resources available to the attacker

Resource constraints play an important role in determining the nature of the attack. An attacker who controls a botnet can use an Internet relay channel and launch distributed denial of service attacks against another node. With respect to attacks where a single node impersonates multiple identities, as in the case of the sybil attack [20] described in section V, IP addresses are also an important resource for the attacker since in DHTs such as Chord [13], the position in the overlay is determined by using a base hash function such as SHA-1 [21] on the node's IP address. The cryptographic puzzles [22] that are sometimes suggested as a way to deter sybil attacks by making the join process harder are futile against an attacker with a botnet and virtually unlimited computation power. Doucer [20] proves that even with the assumption that attackers only have minimum resources at their disposal, it is not possible to defend against them in a pure P2P system.

C. Victim of the attack

The victim of an attack could be an individual node, a particular content or the entire overlay service. If malicious nodes are strategically placed in the overlay, they can block a node from using its services. Attacks could also be launched against specific content [18] or even the entire overlay service. For example, if the malicious nodes are randomly placed in the overlay and drop packets or upload malcontent, then the quality of the overlay would deteriorate.

D. Time of attack

A malicious node could start misbehaving as soon as it enters the overlay or it could follow the rules of the overlay

for a finite amount of time and then attack. The latter could prove to be more harmful if the overlay design suggests accumulating trust in peers based on the amount of time they have been present and/or not misbehaving. In Kademlia [14], for instance, the routing tables are populated with nodes that have been *up* for a certain amount of time. While this provides some robustness from attacks in which the malicious nodes start dropping routing requests from the moment they enter, it would take time for the algorithm to adapt to nodes which start misbehaving in a later stage (i.e., after they have been recorded in routing tables). Similarly for reputation management systems, it is important that they adapt to the current behavior of a peer.

IV. Admission control

Admission control depends on who decides whether or not to admit a node and how this permission is granted. Kim et. al [12] answer these questions independently of any particular environment or application. They define two basic elements for admission in a peer group, a *group charter*, which is an electronic document that specifies the procedure of admission into the overlay, and a *group authority*, which is an entity that can certify group admission. A prospective member first gets a copy of the group charter, satisfies the requirements and approaches the group authority. The group authority then verifies the admission request and grants a group membership certificate.

The group charter and authority verification can be provided by a centralized certificate authority or a trusted third party, or it could be provided by the peers themselves (by voting). The former is more practical and tends to make the certification process simpler although it is in violation of the pure P2P model and exposes the system to attacks typical for server-based solutions (e.g., denial of service attacks targeted to the central authority). The latter, the group authority could either be a fixed number of peers or it could be a dynamic number based on the total membership of the group. The authors argue that even if the group charter requires a prospective member to get votes from peers, the group membership certificate must be issued by a distinct entity. The reason for this is that voters need to accompany their votes with a certificate that proves their own membership. Possible signature schemes that could be used in voting such as plain digital signature, threshold signature and accountable subgroup multisignature are also described. Saxena et. al [23] performed experiments with the different signature schemes and suggest the use of plain signatures for groups of moderate size and where bandwidth is not a concern. For larger groups and where bandwidth is a concern, they suggest threshold signature [24] and multisignature schemes [25].

Another way of handling admission would be to use mechanisms based on trust and recommendation where each new applicant has to be known and vouched for by at least N existing members. The difficulties that such models represent include identity assertion and preventing bot/worm attacks. A compromised node could have a valid certificate identifying a trustworthy peer and it would be difficult to detect this. Possible solutions include sending graphic or logic puzzles easily addressed by humans but hard to solve by computers, also known as CAPTCHA [26].

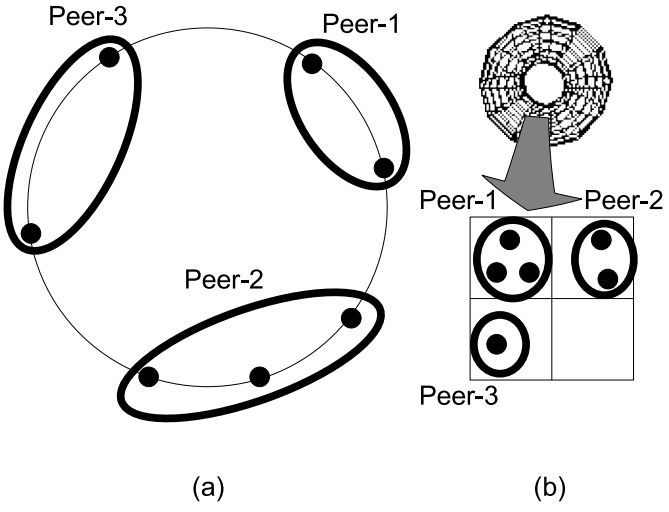

Fig. 1. Geometric representation of key distribution in ring-based (a) and hypercube-based (b) distributed hash tables. In (a) the hash function is unidimensional and thus the key space may be represented as a ring; in (b) the function is bidimensional and thus the space is represented as a torus (whose one only section is detailed).

V. Determining the position in the overlay

For ring based DHT overlays such as Chord [13], Kademlia [14] and Pastry [15], when a node joins the overlay, it uses a numeric identifier (ID) to determine its position in the ring. The positioning of a node determines what information it stores and which nodes it serves. To provide a degree of robustness, content and services are often replicated across multiple nodes. However it is possible for an adversary with sufficient resources to undermine the redundancy deployed in the overlay by representing multiple identities. Such an attack is called a sybil attack [20]. This makes the assignment of IDs very important. One possible scheme to tackle such attacks on the ID mapping is to have a temporal mechanism in which nodes need to re-join the network after some time [27], [28]. Such temporal solutions, however have the drawback that they increase the maintenance traffic and possibly deteriorate the efficiency of caching. Danezis et. al [29] suggest mechanisms to mitigate the effect of sybil attacks by reducing the amount of information received from malicious nodes. Their idea is to vary the nodes used for routing with time and thus avoid a trust bottleneck. Other solutions suggest making the joining process harder by introducing cryptographic puzzles as suggested by Rowaihy et. al [22]. The assumption is that the adversary has limited computational resources which may not be true if the adversary has control over a botnet. Another drawback of such methods is that non-malicious nodes would also have

to perform the extra computations before they can join the overlay.

A possible heuristic to hamper sybil attacks is to employ redundancy at nodes with diametrically opposite IDs (in the DHT ID space) instead of successive IDs as in Chord. The idea behind choosing diametrically opposite nodes is based on the fact that a malicious peer can grant admission to others as its successor without them actually possessing the required IP address (whose hash is adjacent to the former's), and then they can cooperate to control access to that part of the ring. If however admission decisions and redundant content (for robustness), also involve nodes which are the furthest away (diametrically opposite) from a given position, then the adversary would require double resources (IP addresses) to attack. This happens because the adversary would need presence in the overlay at two independent positions in the ring.

Another approach proposed by Yu et al. [30] to limit sybil attacks is based on the usage of the social relations between users. Authors use the fact that as a result of sybil attacks, affected P2P overlays end up containing a large set of sybil nodes connected to the rest of the peers through an irregularly small number of edges. The SybilGuard protocol [30] defines a method that allows to discover such kind of discontinuities in the topology by using a special kind of a verifiable random walk and hence without the need of one node having a global vision of the graph.

It is also worth mentioning that in DHT overlays using different geometric concepts, (e.g., hypercubes instead of rings), peer positions are usually not related to identifiers. In the content addressable network (CAN) [16], for example, the position of an entering node may be either selected by the node itself, or, with little modification to the original algorithm, assigned by peers already in the overlay. However, even when malicious nodes do not know their position before joining, the overlay is still vulnerable to sybil attacks.

VI. Resilience against malicious peers

Making overlays robust against even a small percentage of malicious nodes is difficult [31]. It is therefore important for other peers to identify such nodes and keep track of their number. There are two aspects to this problem. One is the identification itself and the second is the dissemination of this information amongst the peers. Different metrics need to be defined depending on the peer group for the former and reputation management systems are needed for the latter.

A. Identification of malicious peers

For identifying a node as malicious, malicious activity has to be observed first. This could be done in either a proactive way, or a reactive way.

1) Proactive identification: When acting proactively, peers perform periodic operations with the purpose of detecting malicious activity. A malicious node could prevent access to content it is responsible for (e.g., by claiming the object doesn't exist), or return references to content that does not match the original queries [18]. With this approach, publishers of content can later perform lookups for it at periodic intervals and verify the integrity of whatever is returned. Any inconsistencies could then be interpreted as malicious activity. The problem with proactive identification is the management of the overhead it implies: if checks are performed too often, they may actually hinder scalability, while, if they are performed too rarely, they would probably be useless.

2) Reactive identification: In a reactive strategy, the peers perform normal operations and if they happen to detect some malicious activity, then they can label the responsible node as malicious. In a file-sharing application for example, after downloading content from a node, if the peer observes that data does not match its original query it can identify the corresponding node as malicious. Poon et. al [32] suggest a strategy based on the forwarding of queries. If routing is done in an iterative way, then dropping of packets, forwarding to an incorrect node and delay in forwarding arouse suspicion and the corresponding peer is identified as malicious.

B. Reputation management systems

Reputation management systems are used to allow peers to share information about other peers based on their own experience and thus help in making better judgments. Most reputation management systems proposed in the literature [33], [34], [35], [36] are for file-sharing applications. In reputation systems, it should not be possible for a misbehaving peer with low reputation to simply rejoin the network with a different ID and therefore start from a clean slate. To counter this, Kwon et. al [35] store not only the reputation of a peer but also the reputation of files based on file name and content to avoid spreading of a bad file. Another method is to make the reputation of a new peer the minimum possible [36]. Kamvar et. al [36] define five design considerations for reputation management systems;

- Self policing.
- Anonymity.
- No profit to new comers.
- Minimal overhead.
- Robustness to malicious peers.

1) Unstructured reputation management: Unstructured reputation management systems have been proposed by Aydin et. al [33] and Milano et. al [34]. The basic idea of these is that each peer maintains information about its own experience with other peers and resources, and shares it with others on demand. In the system proposed by Aydin et. al [33], each node maintains trust and distrust vectors for every other node that it has interacted with. When reputation information about a peer is required, a node first checks its local database, and if insufficient information is present, it sends a query to its neighbors just as it would when looking up content. However, such an approach requires peers to get reputation information from as many sources as possible; otherwise, malicious nodes may succesfully place targeted attacks returning false values for their victims.

2) Structured reputation management: One of the problems with unstructured reputation management systems is that they either take the feedback from few peers, or if they do from all, then the they incur large traffic overhead. Systems such as those proposed by [35], [36] try to resolve it in a structured manner. The idea of the eigen trust algorithm [36] for example, is transitivity of trust. If a node trusts peer *X* then it would also trust the feedback it gives about other peers. A node builds such information in an iterative way. The algorithm has fast convergence properties [37]. For maintaining this information in a structured way, the authors use a content addressable network (CAN) DHT [16]. The information of each peer is stored and replicated on different peers to provide robustness against malicious nodes. They also suggest favoring peers probabilistically with high trust values instead of doing it deterministically, to allow new peers to slowly develop a reputation. Eventually, they suggest the use of incentives for peers with high reputation values.

VII. Routing and data integrity

Preserving integrity of routing and data, or, in other words, preventing peers from returning corrupt responses to queries and routing through malicious peers, is an important security issue in P2P networks. The data stored on a P2P overlay depends on the applications that are using it. For file-sharing, this data would be the files themselves, their location, and owner information. For real-time communication, this would include user location bindings and other routing information. We describe such data integrity issues separately in Section 7.

A. Data integrity

For file-sharing applications, insertion of wrong content (e.g. files not matching their names or descriptions) or introduction of corrupt data chunks (often referred to as poisoning and pollution) are a significant problem. Bit-Torrent uses voluntary moderators to weed out bogus files and the SHA-1 algorithm to determine the hash of each piece of a file to allow verification of integrity. If a peer detects a bad chunk, it can download that chunk from another peer. With this strategy, different peers download different pieces of a file before the original peer disappears from the network. However, if a malicious peer modifies the pieces that are only available on it and the original peer disappears, then the object distribution will fail [38]. An analysis of Bittorrent in terms of integrity and performance can be found in the work of Pouwelse et. al [10].

B. Routing integrity

To enhance the integrity of routing, it is important to reduce the number of queries forwarded to malicious nodes. Marti et. al [39] developed a system that uses social network information to route queries over trusted nodes. Their algorithm uses trusted nodes to forward queries (if one exists and is closer to the required ID in the ID space). Otherwise they use the regular Chord [13] routing table to forward queries. While their results indicate good average performance, it can not guarantee $\log N$ hops for all cases. Danezis et. al [29] suggest a method for routing in the presence of a large number of sybil nodes. Their method is to ensure that a peer queries a diverse set of nodes and does not place too much trust in a node. Both the above works have been described based on Chord. However, unlike Chord, in DHTs like Pastry [15] and Kademlia [14] there is flexibility in selecting nodes for any row in a peer's routing table. Potentially many nodes have a common ID prefix of a given length and are candidates for routing a given query. To exploit the social network information and still guarantee $\log N$ hops, a peer should select its friends to route a query, but only when they are present in the appropriate row selected by the DHT algorithm.

VIII. Peer-to-peer in real-time communication

The idea of using P2P in real-time communication boils down to distributing centralized entities from conventional architectures over peer-to-peer overlays and thus reducing the costs of deployment and increasing reliability of the different services. Initiatives such as the P2PSIP working group in IETF [40] are working on achieving this by using a DHT for services such as registration, location lookup, and support for NAT traversal, which are normally performed by dedicated servers. Currently, solutions emerged in the working group try to achieve such a distribution adopting three different approaches: P2PP [41] and XPP-PCAN [42] use the overlay only for storing and retrieving user locations (or location of the peers acting as proxies for them), RELOAD [43] and HIP-HOP [44] route the signalling protocol over the mesh of connections between peers, while in ASP [45] peers tunnel both signalling and media flows in end-to-end encrypted connections if it is possible to achieve direct connectivity, or using other peers as relays otherwise.

Even if based on the same technology, overlays used for real-time communication differ from those used for file sharing in at least two aspects:

- Resource consumption. Contrary to file sharing systems where the DHT is used to store huge amounts of data (even if the distributed database is used only for storing file locations, each user usually indexes hundreds or thousands of files), real-time communication overlays only require a subset of the resources available at any given time as users only register a limited number of locations (rarely more than one).
- Confidentiality. While in file sharing applications, where shared files are supposed to be made publicly available, eavesdropping and identity theft do not constitute real threats, in real-time communication, since exchanges of data are usually meant to happen privately, it is essential to have mechanisms to assert identities and to guarantee confidentiality.

In this section we go over the admission issues, and security problems discussed in previous sections, and discuss solutions that would be applicable to real-time communication in P2P.

A. Admission

In order to keep as much compatibility with existing user agents as possible, nodes in P2P communication architectures would probably have to participate as either peers or clients. If a node participates as a client, then it would use the overlay network by simply attaching to a peer or a proxy instead of registering with a server. In most cases users would be able to benefit from the overlay by only acting as clients. However, in order to keep the solution scalable, at some point clients would have to be promoted to peers (admission to the DHT). This requires addressing the following issues:

1) Active vs. passive upgrades: Most existing P2P networks [2], [3], [7], would generally make it the responsibility of clients to determine if and when they would apply for becoming peers. A well known exception to this trend is the Skype network [5], arguably one of the most popular overlay networks used for real-time communications today. Instances of the Skype application are supposed to operate as either super-nodes, directly contributing to the distributed provision of the service, or ordinary-nodes, simply using the service, and the "promotions" are decided by the higher levels of the hierarchy [9]. Even if there is not much difference for a client whether it has to actively ask for authorization to join an overlay, or passively wait for an invitation, the latter approach has some advantages which fit well in overlays where only a subset of the peers is required to provide the service (as in real-time communication):

- An attacker cannot estimate in advance when and if it would be invited to join the overlay as a peer.
- Allows peers to perform long-lasting measurements on sets of candidates, in order to accurately select the most appropriate for upgrading and only invite it when they are "ready" to do so. The opposite approach, that is when clients initiate the join themselves, adds an extra constraint for the peer that has to act upon the request since it doesn't know if and when the peer would attempt to join again.
- Discourages malicious peers from attempting sybil and, more generally, brute force attacks, as only a small ratio of clients has chances to join the overlay (possibly after an accurate examination).

2) When to upgrade: In order to answer this question one would have to define some criteria that would allow to determine the load on a peer and a reasonable threshold. When the load exceeds this threshold, a client is invited to become a peer and share the load. The criteria for determining load can be:

- Number of clients attached.
- Bandwidth usage for DHT maintenance, forwarding requests and responses to and from peers and from the attached clients.
- Memory usage for DHT routing table, DHT neighborhood table, application specific data and information about the attached clients.

3) Which clients to upgrade: Selecting which clients to upgrade would require defining and keeping track of new metrics. The exact set of metrics and how they influence decisions should be the subject of serious analysis and experimentation. These could be based on the following observations:

- Uptime. A peer could easily record the amount of time that it has been maintaining a connection with a client and take it into account when trying to determine whether or not to upgrade it.
- Level of activity. It is reasonable to assume that the more a client uses the service (e.g. making phone calls), the less they would be willing to degrade it.
- Keeping track of history. Peers could record history of the clients they invite and the way they contribute to the overlay.

Other metrics such as public vs. private IP addresses, computation power, and bandwidth should also be taken into account even though they do not necessarily have a direct impact on security.

4) Incentives for clients: Clients need to have incentives for accepting upgrades in order to prevent excessive burden on existing peers. One way to handle this would be to maintain separate incentive management through the use of currency or credits. Another option would involve embedding these incentives inside the protocol itself:

- Peers share with clients only a fraction of their bandwidth (uplink and downlink). This would result in higher latency when using the services of the overlay as a client and better service quality for peers.
- Peers could restrict the number or types of calls that they allow clients to make.

Introducing such incentives, however, may turn out to be somewhat risky. Differences in quality would probably be perceptible for end users who would not always be able to understand the difference between the roles that their user agent is playing in the overlay. Such behavior may therefore be interpreted as arbitrary and make the service look unreliable.

B. Security

1) Targeted denial of service: In addition to bombardment with queries as described in section III, the denial of service attack against an individual node can be conducted in DHTs used for real-time communications if the peers which surround a particular ID are compromised. These peers which act as proxy servers for the victim, can fake the responses from the victim by sending fictitious error messages back to peers trying to establish a session. Danezis et al.'s [29] solution can also provide protection against such attacks as in their solution peers vary the nodes used in queries.

2) Man in the middle attack: The man in the middle attack is well described by Seedorf [46] in the particular case of P2PSIP [40] and consist of an attack that exploits the lack of integrity when routing information. A malicious node could return IP addresses of other malicious nodes when queried for a particular ID. The requesting peer would then establish

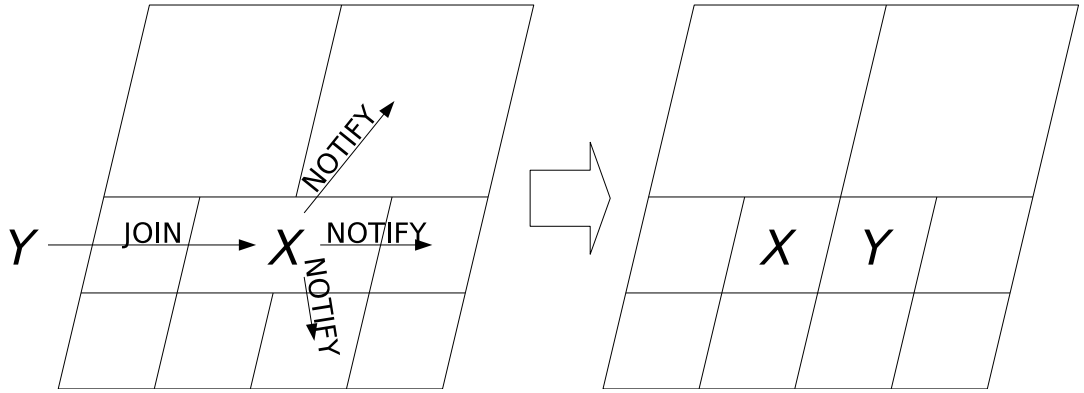

Fig. 2. Node join in an XPP-PCAN [42] overlay: neighbors of the new peer are notified by the admitting peer.

a session with a second malicious node which would again return a "poisoned" reply. This could go on until the TTL expires and the requester gives up the "wild goose chase" [29]. A simple way for entities to verify the correctness of the routing lookup is to employ iterative routing and to check the node-ID of every routing hop that it is returned and it should get closer to the desired ID with every hop. However, this is not a strong check and can be defeated [46].

3) Trust between peers: The effect of malicious peers could be mitigated by introducing the concept of trust within an overlay. This can be done in different ways:

- Using certificates assigned by an external authority. The drawback with this approach is that it requires a centralized element.
- Using certificates reciprocally signed by peers. This mechanism is quite similar to PGP [47]; every peer signs certificates of "friend" peers and trusts any other peer with a certificate signed by one of its friends. However even though it might be theoretically possible, in reality it is extremely difficult to obtain long enough trust chains.
- Spreading the information of each trusted peer to its future neighbors, as shown in Figure 4. This approach, described in [42], works well with some DHTs like CAN, when it is possible to base the trust on some sort of mutual relationship (e.g., neighborhood in CAN [16]).

4) Routing call signalization: One way for implementing real-time communication overlays (as we have mentioned in earlier sections) would be to simply replace centralized entities in signalling protocols like SIP [48] with distributed services. In some cases this might imply reusing existing protocol mechanisms for routing signalling messages. In the case of SIP this would imply regarding peers as SIP proxies. However the design of SIP supposes that such proxies are trusted, and makes it possible for them to fork requests or change their destination, add or remove header fields, act as the remote party, and generally manipulate message content and semantics

However, in a P2P environment where messages may be routed through numerous successive peers, some of which might be compromised, it is important not to treat them as trusted proxies. One way to limit what peers can do is by protecting signalling with some kind of end-to-end encryption, as proposed in ASP [45].

Another option would be to extend existing signalling protocols and modify the way they route messages in order to guarantee secure end-to-end transmission. Gurbani et al. define a similar mechanism for SIP called SIPSEC [49]. It allows nodes to establish a secure channel by sending a CONNECT SIP request, and then tunnel all SIP messages through it, adopting a similar mechanism to the one used for upgrading from HTTP to HTTPS [50].

5) Integrity of location bindings: It is important to ensure that the location that a user registers, usually a (URI, IP) pair, is what is returned to the requesting party. Or the entities that issue the lookup request must be able to verify the integrity of this pair. A pure P2P approach to allow verification of the integrity of location binding information is presented in [51]. The idea is for an entity to choose an asymmetric key pair and hash its public key to generate its URI. The entity then signs its present location with its private key and registers with the quadruple (URI, IP, signature, public key). Any entity which looks up for the URI and receives such a quadruple can then verify its integrity by using the public key and the certificate. Another possible merit of such an approach could be that it is possible to identify the malicious nodes and maintain a black list. However, the resulting URIs are not easy to remember and associate with entities. Discovering these URIs and associating them with entities would therefore require some sort of a directory service. The authors suggest using existing authentication infrastructure for this such as a certified web service using SSL which can publish an "online phone book" mapping users to URIs.

6) Encrypting content: Using P2P overlays for real-time communication implies that content is likely to traverse numerous intermediate peers before reaching its destination. A typical example could be the use of peers as media relays as a way of traversing NATs in VoIP calls.

Contrary to publicly shared files, communication sessions are in most cases expected to be private. It is therefore very important to make sure that no media leaves the client application without being encrypted and securely transported

through a protocol like SRTP [52]. However, the extra processing resources required by the encryption algorithms, the management of keying material (e.g., retrieving public keys when interacting with unknown peers) may constitute an expensive task, especially for mobile devices.

7) Other issues: Identifying more specific threats related to the P2P real-time communications, would require a clearly defined economic model. Answers to the following questions would be helpful.

- To whom do the users pay?
- Do the users only pay when accessing the public telephone network?
- Is the billing done per call or is it fixed?

For instance, the implications of an attack such as taking control over another's user agent or its identity and using it for outbound calls would depend on whether or not this would be economically advantageous for the attacker. Baumann et. al [53] suggests that to prevent unwanted communication costs, gateways for the public telephone network should only be accessible via authenticated servers and dialing authorizations should be enforced. Also it seems that it would be difficult to do billing in a pure P2P manner as it would mean keeping the billing details with untrusted peers.

IX. Conclusion and future work

We have discussed problems in peer to peer security and some solutions at the state of the art, mentioning the suitability and drawbacks of the different schemes. We looked at security keeping the attackers into perspective, their motivation, their restrictions and their targets. Existing security solutions do not seem mature enough to be deployed in pure peer to peer networks. Specifically secure ID assignment and entity-identity association seem the most challenging of problems. Future work on the subject would therefore need to analyse emerging implementations and the way the suggested security solutions perform with them.

Throughout the document we have been insisting that in P2P overlays both signalization and content would have to be encrypted in an end-to-end manner. Further research on the topic would have to investigate possible ways to distribute/exchange the keys necessary for such encryption.

References

[1] "Napster," http://www.napster.com/.

[2] "Kazaa," http://www.kazaa.com/.

[3] "Bittorrent–what is a tracker?" http://support.bittorrent.com/.

[4] S. Androutsellis-Theotokis and D. Spinellis, "A survey of peer-to-peer content distribution technologies," *ACM Comput. Surv.*, vol. 36, no. 4, pp. 335–371, 2004.

[5] "Skype," http://www.skype.com/.

[6] K. Singh and H. Schulzrinne, "Peer-to-peer internet telephony using SIP," *International Workshop, Network and Operating System support for Digital Audio and Video*, Jun. 2005.

[7] "Joost," http://www.joost.com.

[8] "Coolstreaming," http://www.coolstreaming.us.

[9] S. Baset and H. Schulzrinne, "An analysis of the skype peer-to-peer internet telephony protocol," *Infocom*, April 2006.

[10] J. A. Pouwelse, P. Garbacki, D. H. J. Epema, and H. J. Sips, "The bittorent P2P file-sharing system: Measurements and analysis," *4th International Workshop on Peer-to-Peer Systems*, Feb. 2005.

[11] D. S. Wallach, "A survey of peer-to-peer security issues," *International Symposium on Software Security*, pp. 42–57, 2002. [Online]. Available: http://www.cs.rice.edu/ dwallach/pub/tokyo-p2p2002.pdf

[12] Y. Kim, D. Mazzocchi, and G. Tsudik, "Admission control in peer groups," *Second IEEE International Symposium on Network Computing and Applications*, Apr 2003.

[13] I. Stoica, R. Morris, D. Karger, M. F. Kaashoek, and H. Balakrishnan, "Chord: A scalable peer-to-peer lookup service for internet applications," *SIGCOMM*, 2001.

[14] P. Maymounkov and D. Mazi, "Kademlia: A peer-to-peer information system based on the xor metric," *First International Workshop on Peer-to-Peer Systems*, Mar 2002.

[15] A. Rowstron and P. Druschel, "Pastry: Scalable, distributed object location and routing for large-scale peer-to-peer systems," *18th IFIP/ACM International Conference on Distributed Systems Platforms (Middleware 2001)*, Nov 2001.

[16] S. Ratnasamy, P. Francis, M. Handley, R. Karp, and S. Shenker, "A scalable content-addressable network," *SIGCOMM*, 2001.

[17] A. McCue, "Bookie reveals 100,000 cost of denial-of-service extortion attacks," *silicon.com*, 2004. [Online]. Available: http://software.silicon.com/security/0,39024655,39121278,00.htm

[18] E. Sit and R. Morris, "Security considerations for peer-to-peer distributed hash tables," *1st International Workshop on Peer-to-Peer Systems*, March 2002.

[19] J. Liang, R. Kumar, Y. Xi, and K. Ross, "Pollution in p2p file sharing systems," *Infocom*, Mar 2005.

[20] J. R. Douceur, "The sybil attack," *Revised paper, 1st International Workshop. Peer-to-Peer Systems, Lecture Notes in Computer Science*, vol. 2429, Mar 2002.

[21] F. 180-1, "Secure hash standard," *US Department of Commerce / NIST, National Technical Information Service*, Apr 1995.

[22] H. Rowaihy, W. Enck, P. McDaniel, and T. L. Porta, "Limiting sybil attacks in structured peer-to-peer networks," *Technical Report NAS-TR-0017-2005, Network and Security Research Center, Department of Computer Science and Engineering*, Jul 2005.

[23] N. Saxena, G. Tsudik, and J. H. Yi, "Admission control in peer-to-peer: Design and performance evaluation," *Security of Ad Hoc and Sensor Networks*, 2003.

[24] J. Kong, P. Zerfos, H. Luo, S. Lu, and L. Zhang, "Providing robust and ubiquitous security support for MANET," *International Conference on Network Protocols*, Nov 2001.

[25] K. Ohta, S. Micali, and L. Reyzin, "Accountable subgroup multisignatures," *ACM conference on Computer and Communications Security*, Nov 2001.

[26] L. von Ahn, M. Blum, and J. Langford, "Telling humans and computers apart automatically," *Commun. ACM*, vol. 47, no. 2, pp. 56–60, 2004.

[27] T. Condie, V. Kacholia, S. Sankararaman, J. M. Hellerstein, and P. Maniatis, "Maelstorm: Churn as shelter," *University of California Berkeley, Technical report UCB/EECS-2005-11*, Nov 2005.

[28] C. Scheideler, "How to spread adversarial nodes?: Rotate!" *Thirty-Seventh Annual ACM Symposium on Theory of Computing*, May 2005.

[29] G. Danezis, C. Lesniewski-Laas, M. F. Kaashoek, and R. Anderson, "Sybil-resistant DHT routing," *Tenth European Symposium on Research in Computer Security*, vol. 3679, Sep 2005.

[30] H. Yu, M. Kaminsky, P. Gibbons, and A. Flaxman, "SybilGuard: Defending against sybil attacks via social networks," *Proceedings of the ACM SIGCOMM Conference on Computer Communications (SIGCOMM 2006)*, Sep 2006.

[31] M. Castro, P. Druschel, A. Ganesh, A. Rowstron, and D. S. Wallach, "Secure routing for structured peer-to-peer overlay networks," *5th Usenix Symposium on Operating Systems Design and Implementation*, Dec 2002.

[32] W.-K. Poon and R. K. C. Chang, "Robust forwarding in structured peer-to-peer overlay networks," *SIGCOMM*, August 2004.

[33] E. Uzun, M. R. Pariente, and A. A. Selpk, "A reputation-based trust management system for P2P networks," *International Symposium on Cluster Computing and the Grid*, Apr 2004.

[34] E. Damiani, D. C. Vimercati, S. Paraboschi, P. Samarati, and F. Violante, "A reputation-based approach for choosing reliable resources in peer-to-peer networks," *Conference on Computer and Communications Security*, Nov 2002.

[35] S. Y. Lee, O.-H. Kwon, J. Kim, and S. J. Hong, "A reputation management system in structured peer-to-peer networks," *Proceedings*

of the 14th IEEE International Workshops on Enabling Technologies: Infrastructure for Collaborative Enterprise, 2005.

[36] S. D. Kamvar, H. Garcia-Molina, and M. T. Schlosser, "The eigentrust algorithm for reputation management in P2P networks," *12th international conference on World Wide Web*, May 2003.

[37] T. H. Haveliwala and S. D. Kamvar, "The second value eigenvalue of the google matrix," *technical report*, 2003.

[38] X. Zhang, S. Chen, and R. Sandhu, "Enhancing data authenticity and integrity in P2P systems," *Internet Computing*, Sep. 2005.

[39] S. Marti, P. Ganesan, and H. Garcia-Molina, "SPROUT: P2P routing with social networks," *First International Workshop on Peer-to-Peer and Databases*, Mar 2004.

[40] "Peer-to-peer session initiation protocol (P2PSIP) IETF working group." [Online]. Available: http://ietf.org/html.charters/p2psip-charter.html

[41] S. Baset and H. Schulzrinne, "Peer-to-peer protocol (P2PP)," Internet-Draft draft-baset-p2psip-p2pp-00 (Work in Progress), Jul. 2007. [Online]. Available: http://tools.ietf.org/html/draft-baset-p2psip-p2pp-00

[42] E. Marocco and E. Ivov, "XPP extensions for implementing a passive P2PSIP overlay network based on the CAN distributed hash table," Internet-Draft draft-marocco-p2psip-xpp-pcan-00 (Work in Progress), Jun. 2007. [Online]. Available: http://tools.ietf.org/html/draft-marocco-p2psip-xpp-pcan-00

[43] D. Bryan, M. Zangrilli, and B. Lowekamp, "Resource location and discovery," Internet-Draft draft-bryan-p2psip-reload-01 (Work in Progress), Jul. 2007. [Online]. Available: http://tools.ietf.org/html/draft-bryan-p2psip-reload-01

[44] E. Cooper, A. Johnston, and P. Matthews, "A distributed transport function in P2PSIP using HIP for multi-hop overlay routing," Internet-Draft draft-matthews-p2psip-hip-hop-00 (Work in Progress), Jun. 2007. [Online]. Available: http://tools.ietf.org/html/draft-matthews-p2psip-hip-hop-00

[45] C. Jennings, J. Rosenberg, and E. Rescorla, "Address settlement by peer to peer," Internet-Draft draft-jennings-p2psip-asp-00 (Work in Progress), Jul. 2007. [Online]. Available: http://tools.ietf.org/html/draft-jennings-p2psip-asp-00

[46] J. Seedorf, "Security challenges for peer-to-peer SIP," *IEEE Network*, vol. 20, 2006.

[47] P. Zimmermann, "Pretty good privacy: public key encryption for the masses," *Building in big brother: the cryptographic policy debate*, pp. 93–107, 1995.

[48] J. Rosenberg, H. Schulzrinne, G. Camarillo, A. Johnston, J. Peterson, R. Sparks, M. Handley, and E. Schooler, "SIP: Session initiation protocol," RFC 3261 (Draft Standard), Jun. 2002. [Online]. Available: http://www.ietf.org/rfc/rfc3261.txt

[49] V. Gurbani, F. Audet, and D. Willis, "The SIPSEC uniform resource identifier (URI)," Internet-Draft draft-gurbani-sip-sipsec-01 (Work in Progress), Jun. 2007. [Online]. Available: http://tools.ietf.org/html/draft-gurbani-sip-sipsec-01

[50] E. Rescorla, "HTTP over TLS," RFC 2818 (Informational), May 2000. [Online]. Available: http://www.ietf.org/rfc/rfc2818.txt

[51] J. Seedorf, "Using cryptographically generated SIP-URIs to protect the integrity of content in P2P-SIP," *VoIP Security Workshop*, June 2006.

[52] M. Baugher, D. McGrew, M. Naslund, , E. Carrara, and K. Norrman, "The secure real-time transport protocol (SRTP)," RFC 3711 (Draft Standard), Mar. 2004. [Online]. Available: http://www.ietf.org/rfc/rfc3711.txt

[53] R. Baumann, S. Cavin, and S. Schmid, "Voice over IP - security and SPIT," *Swiss Army, FU Br 41, KryptDet Report, University of Berne*, Sept 2006.

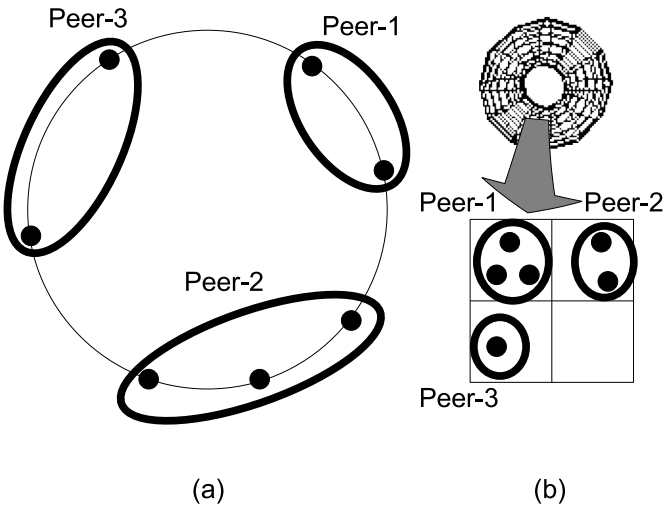

Fig. 3. Geometric representation of key distribution in ring-based (a) and hypercube-based (b) distributed hash tables. In (a) the hash function is unidimensional and thus the key space may be represented as a ring; in (b) the function is bidimensional and thus the space is represented as a torus (whose one only section is detailed).

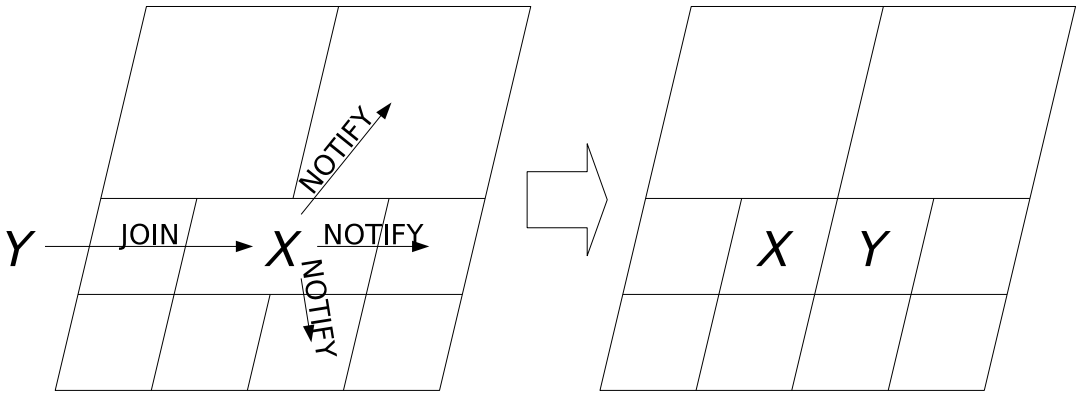

Fig. 4. Node join in an XPP-PCAN [42] overlay: neighbors of the new peer are notified by the admitting peer.