CHALMERS

Growing secure P2P networks

Master of Science Thesis in Networks and Distributed Systems

JONAS BENGTSSON

Department of Computer Science and Engineering
CHALMERS UNIVERSITY OF TECHNOLOGY
UNIVERSITY OF GOTHENBURG
Göteborg, Sweden, May 2010

Growing secure P2P networks

JONAS BENGTSSON

Examiner: DEVDATT DUBHASHI

Department of Computer Science and Engineering
Chalmers University of Technology
SE-412 96 Göteborg
Sweden
Telephone + 46 (0)31-772 1000

Department of Computer Science and Engineering
Göteborg, Sweden May 2010

Abstract

Secure peer-to-peer networks are getting increasingly popular on the Internet. A special type of these networks is the darknet that only uses connections to trusted friends. A major problem is with scaling: connections usually have to be set up manually, which leads to a slow network growth. The goal of this thesis is to investigate if it is possible to use existing social networks between people to set up the connections and grow such networks automatically. As a specific case we consider Freenet, which offers peer-to-peer darknet functionality.

Our work show that it is possible to grow such networks automatically, but that it raises potential security problems. We illustrate this for several social networks.

Contents

1 Introduction

During recent years secure peer-to-peer (P2P) networks have become increasingly popular as a result of various political issues regarding integrity. One type of secure network are the darknets. Such networks are built up by connections that are only between friends. One problem with these kinds of networks are that the connections have to be set up manually. Usually this is done by both parties exchanging a piece of information with each other, typically over the Internet. This information will be called a reference in the remainder of the text.

At the same time as the need for secure P2P networks has increased the usage of social network sites like Facebook have virtually exploded. These have in common with darknets that there exists a list of friends. Assuming some trust exists in these networks, is it possible to use the information about friends at various social networks to set up the darknet connections automatically? The goal of this thesis is to examine and improve our knowledge about this question. It is possible as is proved by three implemented plugins, but it is made hard by the lack of good APIs.

A key idea behind the routing in some darknets is that the network graph has the small world property: that there is only a small number of hops between any pair of nodes. This also seems to hold in various social networks on the Internet. In a case study, we demonstrate this for the GPG web of trust.

This work will in particular focus on Freenet: a P2P network that offers scalable routing in a large darknet. This choice was made because it was the only widely used secure network available at the start of the thesis. Another similar network, OneSwarm, was released a couple of months into the work. Oneswarms methods for exchanging references will be analyzed in detail. Even if this thesis focuses on Freenet the conclusions hold for all darknets and plugins similar to those developed for Freenet could be developed for most darknets.

The outline of the thesis is as follows:

- In Chapter 2, we describe what a secure P2P-networks is. Two main types of networks are discussed.
- In Chapter 3, social networks on the Internet are discussed.
- In Chapter 4, the small world concept is introduced. It is also explained how it relates to social networks and darknets.
- In Chapter 5, we present a case study. The study examines if the small world property exists in the GPG web of trust.
- In Chapter 6, Freenet, the secure P2P network of our study, is described.
- In Chapter 7, existing ways to exchange references in Freenet are explained.
- In Chapter 8, OneSwarm, another secure P2P network is described. It's methods for exchanging references are examined.

- In Chapter 9, the methodology used when developing the plugins is discussed.
- In Chapter 10-12, the three plugins that have been developed are presented. For each plugin there is a description and an evaluation.
- In Chapter 13, we discuss further work. are discussed.
- In Chapter 14, we present the conclusions of this thesis.
- In Appendix A and B, references for two popular secure P2P networks are shown.
- In Appendix C, we investigate how well supported the PubSub and PEP extensions to the XMPP protocol are.

2 Secure P2P-networks

In this section we introduce some terms and concepts that will be used in the rest of this thesis. In particular, what a secure P2P network is. In our context, a secure P2P network is one which somehow allows its participants to hide in the network. We will call the an instance of the software used by a participant in the network for a *node*. An attacker monitoring the traffic between nodes in the network should not be able to determine who accesses or uploads which data. This is usually achieved with tunneling. When the traffic between two nodes is tunneled, it is not sent between them directly, but is instead over a chain of other nodes. This in combination with encryption makes it very hard for an attacker to deduce where observed traffic originated and where it is heading.

In this thesis, we will consider two main types of secure P2P networks, "darknets" and what in Freenet terminology are called "opennets".

2.1 Opennet

In an *opennet*, nodes have connections to strangers. The connections are set up automatically and anyone can connect to the network.

The main problem with these kind of networks are that they are harvestable: an attacker can connect to the network like any normal user would, and map out the structure of the entire network. This can happen if the attacker spends effort in connecting to a large number of stranger. Harvestability makes it easy to block the network in a national firewall. In China, the opennet-part of Freenet is blocked by the Golden Shield (a.k.a. The Great Firewall of China) [29].

Opennets are also vulnerable to certain kinds of attacks. One such attack is the Sybil[1] attack in which an attacker pretends to be several users. This gives the attacker more information about the network than he would have if he had acted as a single user. For example, if the attacker could make all the connections of a

[1]Named after the book Sybil [40]. A true story about a woman with a dissociative identity disorder.

user to be to nodes owned by the attacker, the attacker could easily determine which requests originate at the users node.

2.2 Darknet

A *darknet* is a special type of secure network in which each user is only connected to people they trust, often real-life friends. These connections have to be set up manually. This is done by both users exchanging a piece of information with each other. The information is usually called a *reference* or a key. The reference contains all information needed to set up the secure connection between the friends. Examples of references for two popular P2P-networks, Freenet and OneSwarm can be seen in Appendix A and Appendix B.

A darknet is generally more secure than an opennet. The Sybil attack, that was explained in the opennet section, does not work as well since, while the attacker can create a large cluster of fictionous users, it is hard for the attacker to attach them directly to ordinary users. It is also very hard to harvest a darknet since an attacker only knows the location of his own friends. Many other attacks are also stopped.

A large benefit for security and privacy with a darknet is that it can almost be made invisible. A drawback however is that unlike an opennet there is no way for new users to find users to connect to. It may however still be possible to detect individual users of the network with the use of traffic flow analysis, this however is extremely complex and requires a strong attacker.

3 Social Networks on the Internet

A *social network* is a network that is built up by the relationships between people. Formally it can be represented with a graph where people are nodes and the relationships between people are edges. Social networks have been studied in a number of different fields, such as sociology, computer science and mathematics. Many types of social networks also exist on the Internet, in this section we introduce some of them.

3.1 Social networking sites

A *social networking site* is a web site on which each user has a profile. On the profile, the user can publish various kinds of information. Each user also has a list of friends, which allows various types of interactions to take part. In other words, social networking sites hold a representation of a social network. Some popular social networking sites are described below.

3.1.1 Facebook

Facebook is the social networking site that has the largest amount of unique monthly visitors. Alexa ranks it as the second most popular site in the world and the most popular social network site [6]. It has more than 350 million active users [8]. Facebook was originally called Thefacebook when it was launched in 2004. A year later, the domain name facebook.com was bought for $200,000 [3] and ”The“ was dropped from the name.

3.1.2 OpenSocial

OpenSocial was launched by Google in November 2007 [10]. It is not a social network in itself, but instead an attempt to replace the propitiatory APIs used by each web-based social networking sites with a single set of open APIs. This simplifies for application writers since they only have to learn one API instead of one API per social networking site. A single API also simplifies for the owner of the social network sites since it may allow them to fetch user data, such as profile information and list of friends from other social networking sites that use the same API.

At the moment there are 36 sites that uses that to variable degrees support the OpenSocial API [7]. At the end of 2008, the total number of registered users for the sites add up to about 675 million [12]. The two largest sites are MySpace and the Chinese social networking site 51.com.

3.1.3 MySpace

MySpace is the second largest social networking site in the world [6]. It was launched in 2003. The developers where inspired by Friendster [2], a social networking site that had been launched a year earlier. For several year, MySpace was the worlds largest social networking site, but in 2008 that position was taken over by Facebook [20]. MySpace uses the OpenSocial APIs.

3.1.4 Orkut

Orkut is a social networking site operated by Google that uses the OpenSocial APIs. It was launched in 2004 after having been developed by Google employee Orkut Büyükkökten as an independent project. Orkut has about 80-100 million users globally [11]. It is the most popular social networking site in Brazil, and the second most popular social networking site in India [6]. In the US, Orkut have had some success but it is still less popular than Facebook, MySpace and several other social networking sites. Less than 12% of the Orkut users are from other countries than Brazil, India and the US [25].

3.2 Instant messaging

Besides web sites, there are other technical platforms that allows similar communication between friends. One type is commonly known as an *instant messaging network*, where the communication between friends does not pass through a web site.

Instant messaging networks also represents social networks. Many different instant messaging networks exists but the basic idea is the same for most of them. Each user using an instant messaging network has a list of friends. The user can fetch information about the friends and interact with them in various ways, by sending text messages and files to them for example.

There exists many different clients for accessing instant messaging network. Some clients can only connect to one network, while others can connect to multiple networks. Common clients and protocols are described below.

3.2.1 Pidgin

Pidgin [28], is an open source instant messaging client with support for 16 different instant messaging networks. It was first released in 1999 under the name Gaim. In 2007 the client changed its name after AOL trademarked the acronym "AIM". In the same year, Pidgin was estimated to have more than 3 million users [4].

Pidgin uses the *libpurple* library to handle the different protocols. It is possible to develop plugins for libpurple. The plugins will work both in Pidgin and in other clients that uses the same library.

3.2.2 XMPP

XMPP (Extensible Messaging and Presence Protocol) also known as Jabber, is an open XML-based protocol. As its name suggest it can be used for instant messaging and presence information. At the moment it is used by more than 10 million users, which makes it more widely used than ICQ [17].

For XMPP there is no central authoritative server, as there are for other services like ICQ and MSN. Instead many servers exists and anyone can set up their own XMPP server. Each user belong to a server where they have registered and have been given a *JID* (Jabber Id) on the form username@servername. All communication from a user is to the server he is registered at. The servers communicate with each other, forwarding messages to users that are registered at different servers. This decentralized approach taken by XMPP is very similar to how SMTP is used to deliver e-mail.

Since 2006 Google Talk (GTalk) uses XMPP, both for communicating with clients and with other XMPP servers. talk.google.com is the XMPP server that by far have the highest number of users.

Security is built into the very core of the XMPP protocol by TLS (Transport level security) and SASL (Simple Authentication and Security Layer). Connections from clients to servers can be made over TLS which secures the communication from tampering and eavesdropping. TLS is also commonly used when servers communicate with each other. For authentication SASL can be used. SASL allows for authentication to made without the sending of a password. This is achieved with challenge response scheme.

3.2.3 ICQ

ICQ is an instant messaging program initially released in November 1996 by the Israeli company Mirabilis. The name ICQ is a homophone for "I seek you". In 2001 there was more than 100 million registered ICQ accounts [1]. When a ICQ account is registered, it is assigned an UIN: an unique number. The UINs are assigned sequentially, starting at 1000000. Accounts with low UID are sought after and it occurs that such accounts are sold or stolen for that very reason.

In 1998 Mirablis was bought by AOL. ICQ now uses the OSCAR-protocol, the same protocol that is used by AIM: the AOL Instant Messenger.

3.2.4 Windows Live Messenger

Windows Live Messenger is an instant messaging client created by Microsoft. It was know as MSN Messenger when it first was released in 1999. It uses the *MSNP* (Microsoft Notification Protocol), which is the most popular instant messaging protocol with more than 330 million active users [13]. One drawback with the network is that the traffic by default is unencrypted. There is however a workaround for this if specialized software is used [24].

3.3 E-mail and Other Means of Communication

Besides the networks we discussed above, there are a number of more or less implicit communication networks between people on the Internet. These systems do not necessarily come with an explicit "list of friend", but their usage over time allows one to observe friendships. The use of e-mail is one example of such a network. Even without the use of contact lists, friendships between people e-mailing each other can be observed.

However, in this thesis we shall work mostly with the previous kinds of networks. It can still be noted that our methods and results, could in principle be extended to many other types of networks.

4 Small World Property

The *small world* property is based on the idea that all people in the world are connected together by short chains of friends. The concept was popularized by a famous study by the American social psychologist Stanley Milgram and it has since motivated work in many other disciplines: mathematics, computer science and epidemiology to name a few.

4.1 Six degrees of separation

In the 1960's Stanley Milgram conducted his famous experiment. A number of randomly selected people scattered across the US were each given a packet. They were then told to send the packet to a given stranger, but without using the ordinary postal services. Instead they had to send it to a friend they knew well. The friend would then do the same and send it to a good friend of his and in this way the packet would travel only between friends until it reached its final destination. An example of how a packet could have traveled is shown in Figure 1.

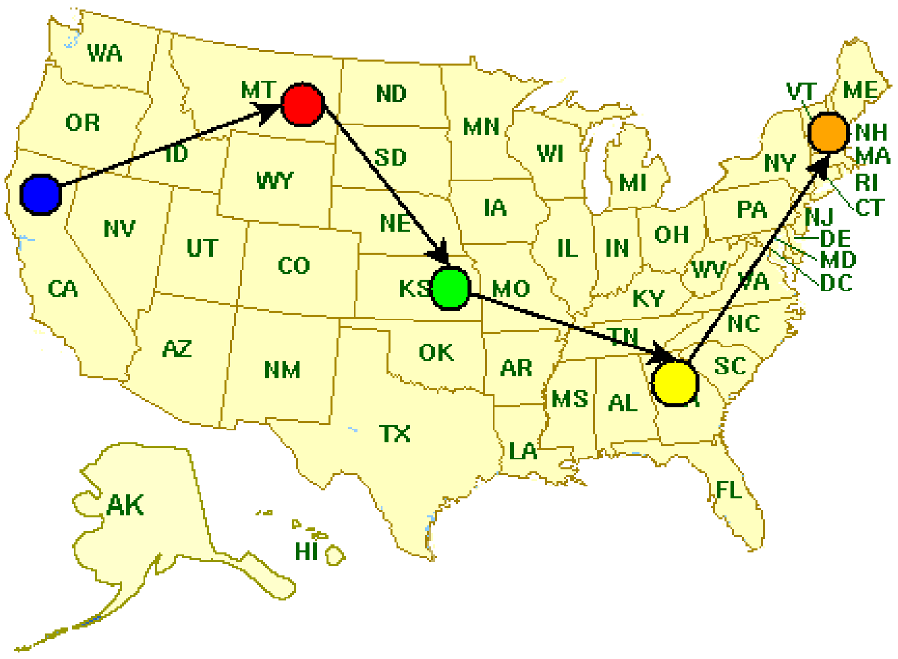


Figure 1: Map over the United States. Each node represents a person and the edges the path a packet is sent over. A packet is sent between the blue and orange node. The length of the path is 4.

Milgram concluded that there was on average *six degrees of separation* between any two people in the US. The term six degrees of separation have since been widely known to the general public even though Milgram never used the term himself.

4.2 Other studies

Milgrams experiment has also been repeated over e-mail and instant messaging and similar conclusions have been reached, however the results have varied. Many studies shows a median value between 5 and 7 [35, 38].

During the initial design of the secure P2P-application OneSwarm, a large scale study was conducted. It investigated the sharing behavior of more than one million users on the music web page last.fm. In the study each user represented a node and the sharing of a song between two users an edge. The average path length between users was found to be 7.1 [36].

4.3 Small worlds in computer science

The small world property have implications on computer science, especially on how routing can be done in secure P2P-network.

Algorithmic work by Jon Kleinberg has shown that routing can be done efficiently in a network with the small world property, even if the nodes in the network only have local information, if greedy heuristics are used [37]. Kleinberg proved this on a network model consisting of nodes placed in a grid with one edge to each neighbor. In the grid shortcut edges were also placed randomly.

Oskar Sandberg expanded on Kleinbergs. He described how routing can be done in more realistic network settings. This was accomplished by using the Metropolis-Hastings algorithm, a randomized algorithm that makes nodes swap places with each other [39]. The work was later introduced as the routing algorithm used by the darknet-part of Freenet.

4.4 Conclusions

Social networks on the Internet and darknets have a lot in common. The small world property exists in many social networks and it also have an impact on how the routing is done in darknets. In a darknet data is only sent over chains of trusted friends, chains that also exist in social networks. This makes it natural to investigate how secure darknets can be set up automatically using information available in social networks.

5 Case Study: GPG

As have been shown in previous sections many social networks exist on the Internet. Studies have also shown that these networks often have the small world property. As a proof of concept and to gain a basic understanding of the phenomena, some experimental simulations were made on the GPG web of trust in order to show that it also has the small world property.

The choice fell on GPG because of its open nature and its heavy use of cryptography. Since it is mostly used by advanced users concerned about computer security the quality of the graph is also very good, and should reflect some degree of trust between users. The whole graph of the network is also widely available.

5.1 GPG

GPG (Gnu Privacy Guard) [31] is an open system for encrypting and signing e-mails. Each user has a list of keys they trust where each key belong to a different user. The users and their trust values form a graph where each node corresponds to a user and where each trusted key represents a directed edge. The edges are directed since it is possible that user A trusts user B but that user B does not trust user A. This graph is called a *web of trust* and can be seen as a social network.

5.2 Wotsap

Wotsap (Web of trust statistics and path finder) [32] is a tool for finding paths and various kinds of information in the GPG web of trust. The program is bundled with a database file containing the web of trust. It is the data extracted from the Wotsap database file that have been used for the simulations, see below.

5.3 The Floyd-Warshall algorithm

The *Floyd-Warshall* [34] algorithm is utilized for finding the shortest path in weighted, directed graphs. The algorithm uses dynamic programming and is able to find the shortest paths between all pairs of nodes in a single execution.

The algorithm uses $O(n^2)$ space and $O(n^3)$ time where n is the number of nodes in the graph.

5.4 Experiment methodology

The Wotsap graph extracted on 23 January 2009 containing 41309 nodes was used for the simulation. Even though the GPG web of trust is directed it is also interesting to look at the undirected version of the graph[2], since friendship relations in real life usually are undirected.

The algorithm was implemented in C. This yielded fast execution times and allowed fine grained control of the memory usage.

Three loops each executed 41309 times. It took two days to execute on an AMD Athlon64 X2 6000+ 3.0GHz processor, using only one core.

Since the Floyd-Warshall algorithm is dynamic, a lot of memory was required. The assumption that no path is longer than 256 hops was made. This allowed each distance to be stored in a single byte. An array consisting of 41309 columns and 41309 rows of bytes had to be allocated which required a total of 1627 MB of memory.

5.5 Results

Both the directed and undirected version of the graph have the small world property. For all node pairs the most common hop-length is five for both graphs. The average hop-length is 5.96 for the directed and 5.05 for the undirected graph.

[2]An undirected graph can be built by always having one edge in each direction between two nodes. If there is only one edge between two nodes a second edge is added.

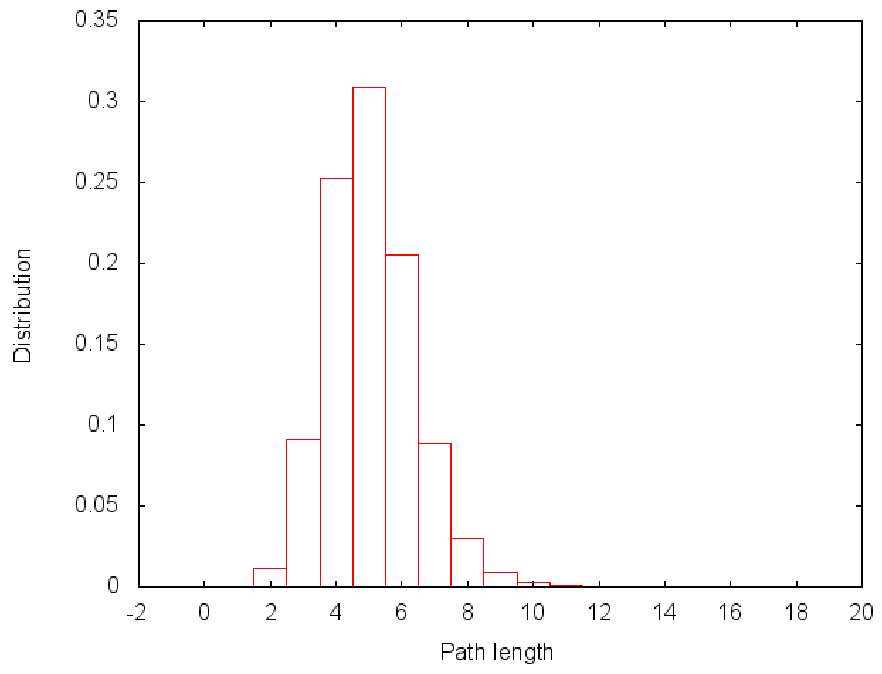


Figure 2: The distribution of path lengths between node pairs in the undirected graph. The most common path length is 5, which occurs for more than 30% of all node pairs.

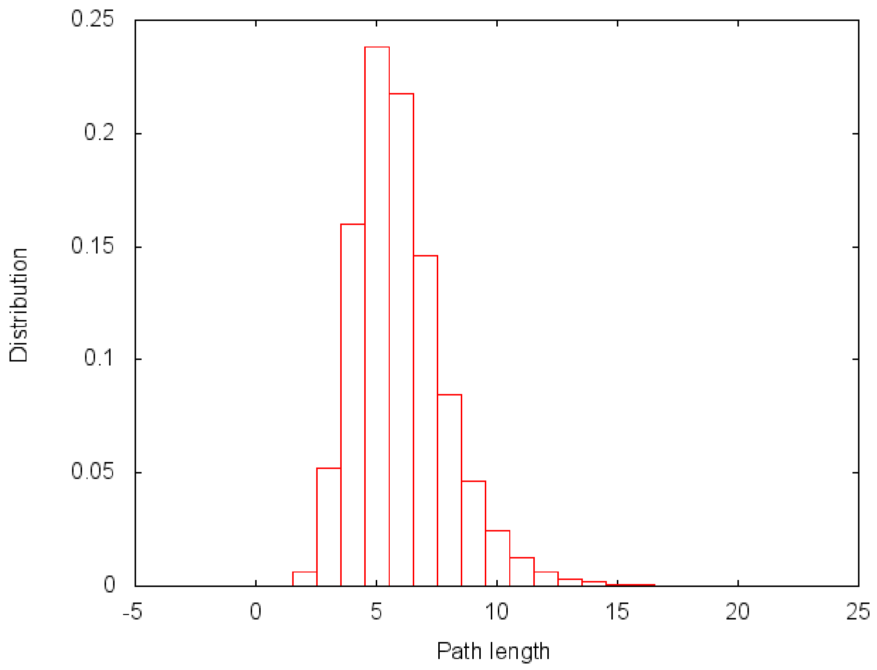


Figure 3: The distribution of path lengths between node pairs in the directed graph. The most common path length is 5, which occurs for almost 25% of all node pairs.

6 Freenet

Freenet [33] is a secure P2P network that will be used in this thesis. The main goal behind Freenet is not file sharing but instead to provide freedom of speech [30]. Freenet is written in Java which allows it to run on many different computer and operating systems.

A computer running an instance of the Freenet software, is called a node. Each node has a number of encrypted connections to other nodes.

There are two different kinds of network modes supported by Freenet, opennet and darknet. A node can either be part of both types of networks, or only one of them. Here, we will mostly consider darknet connections

Freenet is different from most other P2P networks since it uses a *distributed storage*: each node uses a certain size of the hard drive as a datastore. The network acts as a gigantic hard drive were content uploaded is scattered across the network. This is important from a security point of view, the nodes storing the data is not the one that uploaded it. If a user that has uploaded sensitive data is arrested and his node seized, that data will still remain on Freenet. Uploaded data is also encrypted which makes it hard, but not impossible for the owner of a node to determine which data is stored locally. This provides *plausible deniability* of what kind of data is stored.

Freenet allows much more than traditional file sharing. It is possible to publish pages on Freenet similar to a websites, these are called *freesites*. It is also possible to build different types of applications on top of Freenet. Examples of this are e-mail (*Freemail*), and messaging boards like *Freetalk*, *Frost*, and *FMS*.

6.1 Opennet

Originally Freenet was an opennet only. A file containing references for special seed nodes is included with Freenet. When a user starts his node, the node first connects to seed nodes. The seed nodes then help the newly started node to find other nodes.

Every node has a location, a number between 0 and 1 which remains fixed. Each block of a file also has a location. If the network graph has the small world property, it should be possible to find data in $O(\log(n)^2)$ hops.

6.2 Darknet

A darknet was introduced into Freenet with the 0.7-release in May 2008. In the darknet the node locations are not fixed. Instead darknet peers tries to swap location with each other using the randomized Metropolis-Hastings algorithm. Routing is done in a way similar to how it is done in the opennet. However, in order for the routing to be efficient in the darknet the network graph has to have the small world property. This makes it important that the users only add connections to friends, and not to strangers.

6.3 Freenet reference

The reference is the piece of information that two Freenet nodes have to exchange in order to establish a secure connection. It can be seen in Appendix A.

The reference contains the cryptographic information needed to set up a secure communication channel between the two peers. It also contains the IP-addresses and port numbers that should be used. A location on the network where the nodes can publish their IP-address if it changes is also included.

6.4 FProxy

FProxy is a web based interface to Freenet. It allows the Freenet node to be controlled by a web browser. For a local node the standard URL to use is http://127.0.0.1:8888. A benefit of having a browser-based interface is that freesites can be visited in the same way ordinary websites are. FProxy is needed by the work described in Chapter 10 and Chapter 11.

6.5 FCP

There exists a protocol, *FCP* (Freenet Client Protocol), that allows external applications to communicate with the Freenet node. The external program uses a TCP-socket and connects to the default port of 9481. FCP is used by the plugin described in Chapter 12.

6.6 Plugin architecture

It is possible to develop plugins for Freenet. Plugins have to be written in Java and be packaged in jar-files. The manifest of the jar-file contains a line telling Freenet which class to use as the main class. If a plugin uses external libraries they have to be included in the jar-file.

Plugins can be loaded and visited by a web browser connection to FProxy.

7 Existing Ways to Exchange References in Freenet

Before we develop methods to exchange references in Freenet, we study existing manual methods for exchanging references. At the moment no automatic methods are available. A couple of common methods are described below.

7.1 Existing Communication Channel

A common way to exchange references is to manually send it over an existing communication channel to a friend. It could for example be done over e-mail, instant messaging, or over a social networking site like Facebook. This however, is only as secure as the underlying communication channel.

The main problem is that the user has to know in advance if some friend uses Freenet, or the friends receiving the references could be annoyed or confused. It also requires manual work, both for sending references and for adding the references friends send in return.

7.2 Freenet Messaging Boards

Several messaging boards exists that are build on top of Freenet. One of them is *Frost*. Frost suffers from regular DOS-attacks. It is very hard to stop these attacks since anyone can post and it is not possible to trace the attacker.

There also exists two other messaging boards, FMS (Freenet Message System) and FreeTalk. They do not suffer from DOS-attacks, since they are based on the concept of a web of trust. However, this is actually a form of community censorship, the users of the systems can give trust values to each other. This allows people with views disliked by the majority of users, to be silenced.

Boards exists within these programs that allows people to exchange references. When using such a board it is very important to use it with a new identity, an identity that was generated only for that purpose.

The reference itself should not be posted since it contains sensitive information like the IP-address of the user. Instead a request should be made and after a reply the two users should exchange references in a private message.

Exchanging reference with a total stranger is not a good idea. There have been attempts where users asks people with a similar interest to exchange references with, an attempt to keep the small world property.

7.3 #freenet-refs

For some time the IRC channel #freenet-refs at irc.freenode.net existed. It allowed strangers to exchange references with each other. Exchanging references in this way has several problems. The most obvious problems are security related. Connecting to a stranger found with IRC is no better than finding a stranger on opennet. The largest problem however was that it lead to poor network topology. In order for the routing to work efficiently the darknet graph has to have the small world property.

Even if users in the channel does not publish their references anyone observing the channel could map out the relationship between them. It might also be possible to find out the IP-addresses of the users in the channel depending on the settings of the IRC-server.

The channel no longer exists because of the many issues related to security and network topology.

7.4 Freenet-Cards

Visiting cards that people use to exchange contact information have been created that contains a Freenet reference [21]. If two people meet in real life they can exchange cards. They could then enter the reference on the card manually. Since the reference is quite long this is hard and error prone. Some alternatives have been proposed where the cards instead of containing the reference itself contains a way to find the reference, like an URL.

8 OneSwarm

The work done for this thesis focuses on Freenet, but there is another major secure P2P-network in existence that aims to provide privacy, *OneSwarm*.

This section will briefly describe what OneSwarm is and what major differences it has compared to Freenet. How references are exchanged will also be examined in detail. Unlike Freenet, OneSwarm by default includes several different ways for friends to exchange references. Before we make our own software for this it is helpful to examine this as an existing solution.

8.1 Description

OneSwarm was developed at the University of Washington and released in April 2009, after the work on this thesis had begun. It is based on the BitTorrent client *vuze* (formerly know as azureus). It was developed with file sharing in mind [36] and allows downloads to be made both from insecure BitTorrent trackers and from the secure OneSwarm network with the same client.

Unlike Freenet, OneSwarm was originally a darknet only. In order for it to work references had to be exchanged, either manually or with the tools included. The references used by OneSwarm are a lot shorter than the ones used by Freenet. An example of an OneSwarm reference can be seen in Appendix B.

OneSwarm doesn't have a distributed storage, instead it uses a more classical file sharing approach in which each user has a list of files that are stored locally. Searches in the network are done with flooding, that is queries are sent to all friends, which means that the search functionality of the network doesn't scale.

OneSwarm uses a *distributed hash table* which is used by clients to find the latest IP-address and port number of their friends. A consequence of using the hash table is that users can't hide that they are a part of the network from their ISPs, even though OneSwarm is a darknet. This is because all connections to the hash table are done to a static hostname and port number.

In OneSwarm it is possible to add friends as *limited*. Friends added as limited can't list which files that are shared by the given user. The idea behind this is to allow users to assign different levels of trust to different friends.

8.2 Ways to exchange references

By default OneSwarm includes five different ways for friends to exchange references. References could be added manually, in a way similar to how it is done in Freenet. There is also functionality for adding friends at the local network automatically by using broadcasting. There are also three more sophisticated ways which will be describe in detail.

8.2.1 GTalk

It is possible for users to exchange reference with friends that have a GTalk account. All GTalk traffic passes through googles XMPP servers. These servers however doesn't have support for storing personal information such as references. In order to accomplish the exchange of reference the developers of OneSwarm set up a number of dedicated computers at the University of Washington. References are not sent between friends that want to exchange them, but instead to and from these centralized servers where they are stored.

There is a form in the OneSwarm user interface which allows the user to enter his GTalk username and password. After the user has done this, OneSwarm logs in on the GTalk account at the XMPP-server and sends the reference to special GTalk accounts that are connected to the centralized servers. The servers then stores the reference together with hashes of the usernames of the users friends. If a reference for a friend which username matches the hash is stored at the server it is sent back to the user and added automatically.

This approach makes it easy for friends to exchange references if both have a GTalk account, but there are several major problems related with it. The main one is about security. It is possible for the servers to be attacked which may cause stored references and username hashes to be modified, which would open up for man-in-the-middle attacks. Since the solution is centralized there are also reliability issues; if the servers are down, the service won't work.

8.2.2 Community Servers

It is possible for users to subscribe on potential friends from what are called *community servers*. These friends are added as limited which doesn't allow them to browse shared files.

There is no restriction on how can set up a community server. A list of which community servers are in use can be managed in the OneSwarm user interface. There is support for community servers with authentication. Such servers requires the users to log in with username and password. This can be used to ensure that the users are in fact related to each other in some way, as members of the same community for example. If community servers are used without authentication, which is the most common way, the "friends" received by a server are actually total strangers. This largely defeats the purpose of OneSwarm being a darknet and practically makes it into an opennet.

8.2.3 E-mail invitations

OneSwarm has support for invitations that are sent over e-mail. When an invitation is created a special invitations code is generated. The invitation codes for all created invitations are stored and can be managed by the user that created them. The invitation e-mail that can be sent when an invitation is created consists of a description of how OneSwarm can be downloaded and installed together with a URL used for accepting the invitation. The invitation code is embedded in the URL and when the URL is accessed OneSwarm uses the code to look up the IP-address of the friend who sent the invitation in the hash table. A connection is then established and references are exchanged.

The advantage with this approach over simply sending a reference in an e-mail is that the only step that has to be taken in order to set up a secure connection is for the receiver of the invitation to accept it. If a reference was sent instead the receiver would not only have to add it, but would also have to send his reference to the user he received the reference from.

8.2.4 Evaluation

We consider GTalk and community servers to be the most interesting of the ways to exchange references. The usage of e-mail invitations and support for adding users on the local network are interesting, but of limited value. There are however major problems with both GTalk and community servers. Both are centralized solutions which is not always acceptable.

9 Methodology

One of the goals behind the thesis was to write software that grows secure P2P networks by using existing social networks. This requires us to use some means of interaction. In order to do this it was necessary to have a technical understanding of how different social networks work. The technical details about several social networks, including *application programming interfaces* (APIs), were examined to this end.

In order to chose suitable networks, a list of criteria related to security were defined, criteria that the software being developed ideally should have.

- The software should not use a dedicated server, instead it should use the servers of the social network. The reference could either be sent over the network to friends, or stored in the network for later retrieval by friends.
- If the reference is stored in the network it should be possible to control who can see the it, since privacy is usually of importance in a darknet. No one else than the owner of a reference should be able to modify it.
- It should not be possible for strangers to see if the user uses the software. If a user uses the software it is also highly probable that the user is part of a darknet as well, which the user may not want to be public knowledge.

- The connections to the social network should be secure. This is not only of importance when sending the reference over the network, but also when sending the reference to the network for storage, or when a friend retrieves the reference from the network.

The properties of many different social networks was compared against the criteria defined above, both for social networking sites and for instant messaging networks. In this section we present methods and results.

9.1 Social Networking Sites

For almost all social networking web sites, there exists a standardized API that allows developers to create applications that interact with the site.

During the search for a suitable social networking web site, all major APIs were examined and their properties were compared against the criteria defined above.

9.2 Instant Messaging

For instant messaging networks, there usually does not exists a single well defined API. Instead many different instant messaging clients and libraries exists that provides APIs. The available open APIs were examined and compared against the criteria defined above.

9.3 Results

We now discuss the results we got after comparing the APIs of various social networks against the criteria we defined earlier.

9.3.1 Social Networking Sites

All of the major social networking web sites were examined, but none of them fulfilled all of the criteria. Almost all of the APIs assumed that the software is web based, that is, it is run on a remote web server. Only two APIs allowed applications to be run locally on an end-client, Facebook and OpenSocial.

Facebook allows references to be stored at the Facebook servers in the form of a text note. These notes are visible in the users profile and are usually used for diaries or other means of direct communication. There is also good support for access control, but it is not well defined if the connections are secure. A plugin for Facebook was developed and is described in Chapter 10.

A lot of effort was spent with getting it to work with OpenSocial, an API that is used by many different social networking sites. Since the OpenSocial API is open and widely used, we believed that it would work for at least one of the many social networking sites that use it. It was not possible however, mostly because the lack of good access control.

9.3.2 Instant Messaging Networks

There exist libraries that allow many different instant messaging networks to be accessed by applications with a single API. Since the networks can be quite different, only the most common task like sending text messages are supported. A plugin was developed that uses this kind of library. It is described in Chapter 12.

More specialized APIs, focusing only on a single protocols exists. This allows a program to use special functionality available in a specific protocol. A plugin that uses the publish/subscribe-functionality in XMPP was developed. The plugin is described in Chapter 11.

10 Facebook Plugin

In this section we describe the design, implementation and evaluation of the Facebook plugin. At Facebook, each user has a *profile*. The profile is similar to a homepage and allows the user to publish various kinds of information. Each user also have a list of friends and can optionally be a member of *networks*, groups of people sharing interests. There are four main type of interactions between friends on Facebook:

- A *wall post* is a text message that a user can post on a profile, either his own or that of a friend. Wall posts are usually short and of an informal nature. Only the most recent wall posts are displayed on the profile. It is possible to set permissions for which users should be able to see which posts.
- *Notifications* are private text messages that can be sent between friends.
- A *note* consists of a title and a body of text. The content of a note can be very long. It is possible both to set permissions on individual notes and default permissions for all newly created notes.
- A part of the Facebook API is the *Data Store API*, an API that allows applications to store data at the Facebook servers, data not visible in the profile. The API lacks user level permissions and is of the time of writing disabled.

The idea behind the plugin was to use one of these types of interactions to share references. Of the interaction types, notes seemed most suitable. The advantage with notes over notifications is that only one note has to be published in order for a reference to be shared with all friends. With notifications a notification has to be sent to every single friend. The notification approach is also not transparent to the friends since they would receive the cryptic looking reference in their notifications inbox, even if they are not interested in being a part of a darknet.

At the start of the thesis the Facebook API was very limited. There was support for posting at walls and for sending notifications. However, there was no support for reading posts from the wall and only the last received notification could be read. API support for notes was lacking all together. This lead to the development of a plugin based on web scraping. Later improvements to the API allowed a new improved plugin to be written that doesn't use web scraping.

Both plugins are written in Java and are run as plugins in FProxy. The process of authenticating a user at Facebook differs, but the main interface of the plugins is the same and can be seen in Figure 4 and Figure 5. After the user has authenticated himself, the reference is published as a note in the users profile. The profiles of the friends are then scanned for notes containing references. If new references are discovered, the user is asked if the references should be added. After a reference has been added a notification is sent to the friend that owns it, which allows the friend to add the reference of the user. It is also possible to send invitations to users, notifications that contains instructions on how to install Freenet and the plugin.

Both plugins are described and the plugin specific details are evaluated separately. Finally issues regarding using Facebook for exchanging references in general will be discussed, such as the quality of the Facebook graph and the Facebook terms of service.

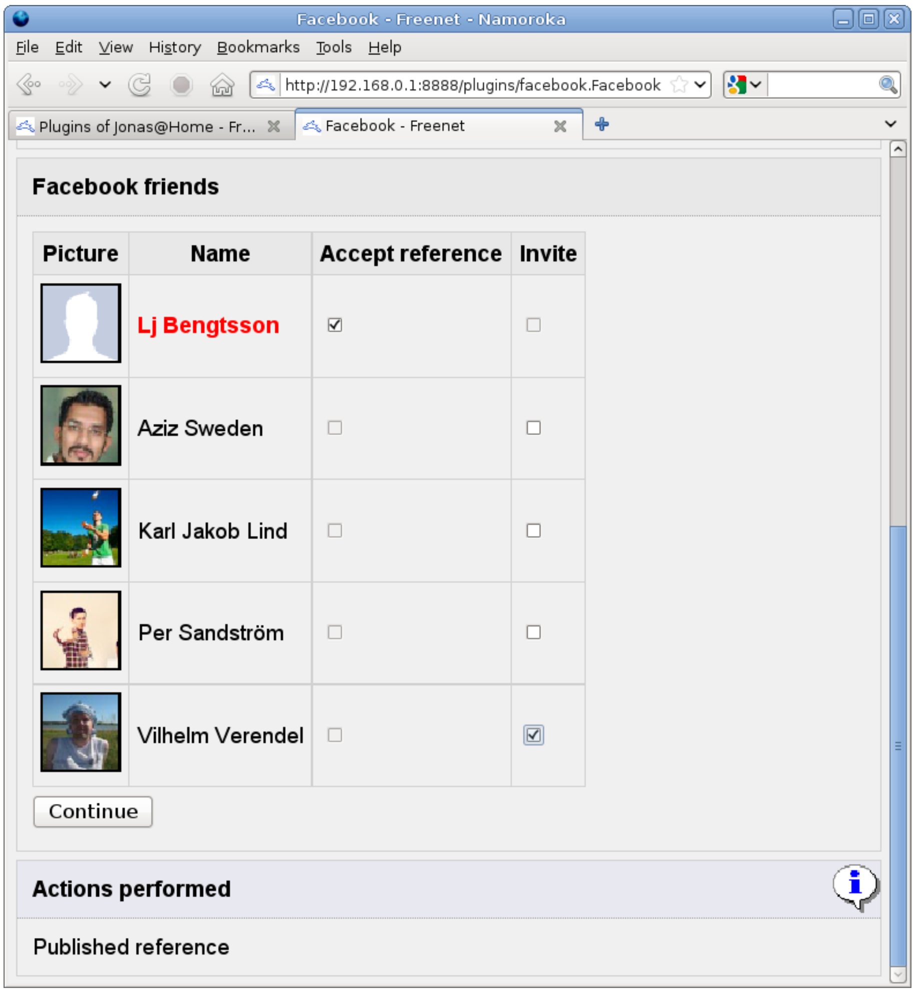
10.1 Plugin using web scraping

Since the Facebook API was too limited, the only remaining option was the technique commonly known as *web scraping*. Web scraping is an inelegant way to extract information from web sites. A web scraping program visits a web site in a similar way a person would, but in an automated manner. It downloads the various pages, follows links, fills in forms and extracts information from the pages. There exist a Facebook chat plugin [27] for Pidgin which extracts and posts data to Facebook with the use of web scraping. This plugin was used as inspiration. For downloading pages from Facebook the HttpComponents [22] library from the Apache foundation was used. The library has support for cookies which allows the plugin to keep state after it has filled in the login form. To extract data from downloaded pages we used regular expressions.

For the authentication process, the plugin simply asks the user to enter an e-mail address registered with a Facebook account and a password.

10.1.1 Evaluation

There are several problems with a plugin that uses web scraping. The main problem is that the plugin may cease to work if the layout of the Facebook site changes. If the plugin is to work in the future it would require regular maintenance. At the time of writing the plugin does not work because of this. Screen scraping also generates a lot of traffic. The average Facebook has 130 friends [8]. Since the plugin makes two HTTP GET request for each friend this would add up to a total of 260 HTTP GET requests. With the Facebook API it is possible to make batch requests, which combines multiple individual operations into a single request. This allows less bandwidth to be used and can result in lower latency [16]. Something similar is not possible with the standard HTTP that is used for the web scraping. Web scraping is also quite unpopular and it might even be against the Facebook terms of service.



Facebook - Freenet - Namoroka
File Edit View History Bookmarks Tools Help
http://192.168.0.1:8888/plugins/facebook.Facebook
Plugins of Jonas@Home - Fr...
Facebook - Freenet

Facebook friends

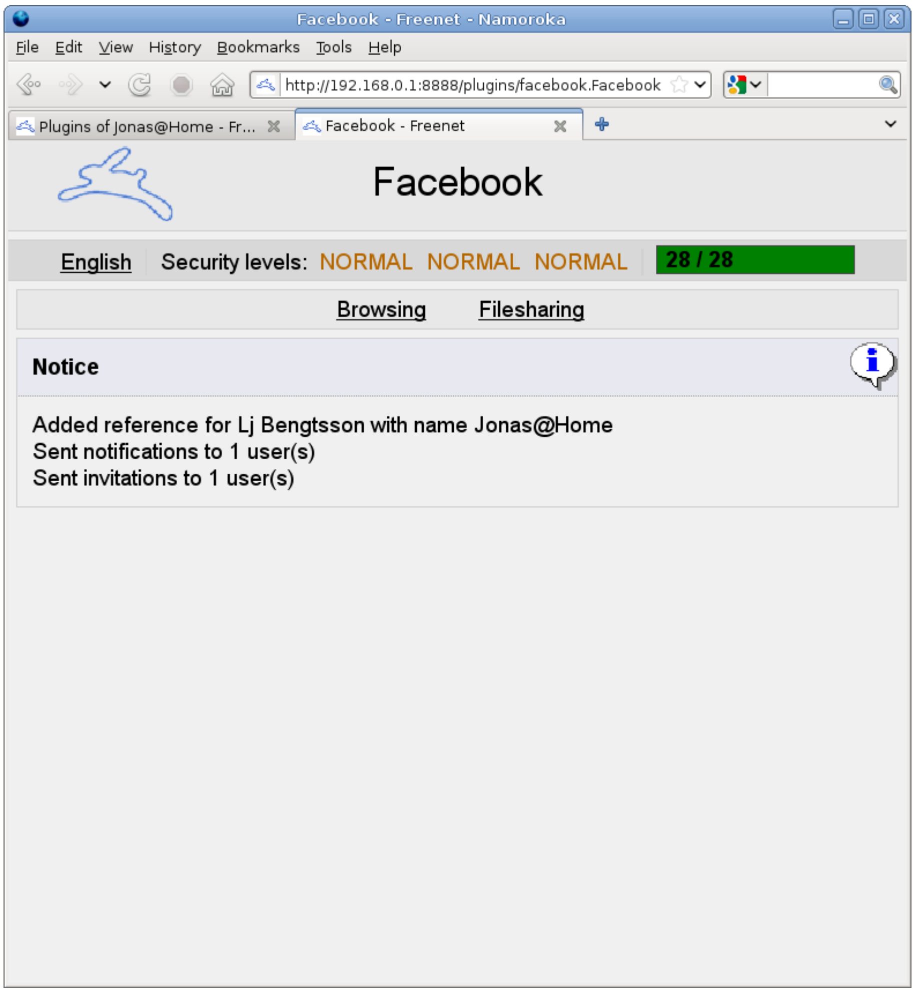
Picture	Name	Accept reference	Invite
	Lj Bengtsson	☑	☐
	Aziz Sweden	☐	☐
	Karl Jakob Lind	☐	☐
	Per Sandström	☐	☐
	Vilhelm Verendel	☐	☑

Continue

Actions performed

Published reference

Figure 4: User interface for the Facebook plugin after the user has logged in. The interface has a list of friends and checkboxes for accepting published references and for sending invitation messages. After pressing the Continue-button, the selected actions will be executed and the user will be taken to the page shown in Figure 5.


Facebook - Freenet - Namoroka
File Edit View History Bookmarks Tools Help
http://192.168.0.1:8888/plugins/facebook.Facebook
Plugins of Jonas@Home - Fr...
Facebook - Freenet
Facebook
English
Security levels: NORMAL NORMAL NORMAL
28 / 28
Browsing
Filesharing
Notice

Added reference for Lj Bengtsson with name Jonas@Home
Sent notifications to 1 user(s)
Sent invitations to 1 user(s)

Figure 5: User interface for the Facebook plugin after the user has pressed the Continue-button in Figure 4. Status about the selected actions are displayed.

The large flexibility with web scraping has several benefits from a security perspective. It is easy to force all connections to the Facebook server to be over SSL/TLS. The permissions on the notes can be set to appropriate values, something that is not possible to do with the API. It it also hard to differentiate between a connection from a web scraping program from that of an ordinary user. If the API was used over an insecure connection it would be easy for an attacker to see which application the connection was a part of. A web scraping plugin also doesn't require to be installed at the Facebook web site.

10.2 Plugin using the Facebook API

After the web scraping plugin was finished, Facebook released some improvements to the Facebook API. The largest improvements was the addition of an API for publishing and reading notes. This allowed for an improved plugin to be developed [14]. We used the Java Facebook API library [19] which communicates with the Facebook servers over a REST-like protocol [18][3]. The Java Facebook library lacked support for the new notes API and thus it had to be modified.

10.2.1 Application types

There are two types of applications that can access the Facebook API, desktop and server applications. *Server applications* are the most common. A server application is hosted at a web server and can be embedded in the profile of the user. A *desktop application* is an application that is run as a normal program at a desktop computer. The plugin is run as a desktop application.

10.2.2 Authentication

The web scraping plugin needs the user to supply his or her login information. This is however not supported by the Facebook API. Instead it uses an authentication scheme in which the user never has to enter his password. The plugin generates a URL at Facebook which contains a token and an application identifier and asks the user to visit it. The user visits the URL and logs in to Facebook if he is not already logged in. Facebook then remembers the token and application identifier. The plugin then uses the token and an application key to generate a session key. The session key is the used for all future requests of the session.

User interface-wise this authentication approach works very well together with the plugin since both the use of the plugin and the authentication at Facebook can be done in the same browser. The URL that has been generated is simply presented to the user as a link that opens in a new browser window.

There is one serious problem with this approach for a desktop application, the application key can usually not be kept secret if dedicated authentication server

[3]The client makes method calls over the Internet by sending HTTP requests to the Facebook API REST server at http://api.facebook.com/restserver.php

is not used. Such a server could store the application key and return session keys when given a token. This approach was not taken because of its centralized nature, it doesn't scale and provides a single point of failure. Instead the application secret is included in the program, an approach which is quite common for desktop applications. This doesn't allow Facebook to authenticate the requests from the plugin. It is however, still possible to have a secure connection if SSL/TLS-connections are used.

10.2.3 FQL

There is a limit on how many request an application using the Facebook API can do. An early version of the plugin looped through all friends and made requests using the notes API for each friend. However, for users with many friends this caused the plugin to reach an upper limit on the number of request that Facebook has set. This was solved with the use of *FQL* (Facebook Query Language), a language modelled after *SQL* (Structured Query Language): a language that is commonly used to extract data from relational databases. FQL allows data be extracted from Facebook in a way similar to how SQL can be used to extract data from a relationship database. This provides a concise way to define even complex data queries which can reduce the number of requests that has to be sent. Our implementation thus uses FQL to scale with the number of friends.

10.2.4 Evaluation

The plugin that uses the API solves many problems that faced the web scraping plugin. Because the use of FQL, the plugin requires less bandwidth and is more responsive than the screen scraping plugin. Since it uses the official API the need for maintenance is also low. Changes to the Facebook site won't affect the API. The plugin only requires maintenance if major modifications are done to the API by Facebook. The main problems with the plugin are instead related to security. The API does not allow applications to set permissions on newly created notes with the API. Instead, the plugin prints a text message, telling the user to set the permissions manually. If the user doesn't follow this advice the reference might be visible to people that are not friends, depending on settings in the users profile. It is also not well known which security implications the practise of bundling the application secret with the application has. The practise is often condemned but still widespread.

10.3 Quality of social graph

The average Facebook user has 130 friends [8]. This is probably more than a normal person would consider in real life. This means that most people are friends on Facebook with people they are not friends with in real life. When using the plugin it is important to only add references by users that they trust. This is why none of the plugins adds references automatically but instead asks the user to confirm the process. Even if the plugins allows the user to choose which users to add, most ordinary users are not aware that the Freenet darknet should have the small world property and will probably try to increase to number of darknet connections by adding as many Facebook friends as possible.

10.4 Can we trust Facebook?

Using Facebook to exchange references requires a large degree of trust in Facebook, both in technical platform and in the Facebook company. All references uploaded by the program are stored at Facebook servers and could theoretically be both viewed and modified by employees at Facebook, there is also the possibility that an outside attacker attacks Facebook.

In February 2009 Facebook changed their terms of service. The new terms allowed them to store all images and text, including references, that a user has on his profile indefinitely, even if the account of the user has been removed [9]. After large protests, including the establishment of a Facebook group [26] which quickly got more than 100,000 members, Facebook changed their terms of service again.

10.5 Evaluation

Using Facebook to exchange darknet references can be suitable for use in countries were darknets are not illegal, like Sweden. Using one of the plugins developed allows the darknet to grow rapidly.

It is not recommended that the plugins are used were using a darknet is illegal. There is always the theoretical possibility that Facebook will collaborate with the legislation and government of such countries.

Both of the developed plugins have their respective advantages and disadvantages. Security wise the web scraping plugin is better but has several problems such as high amount of network traffic and problems with maintenance. The plugin that uses the API solves the problems with traffic and maintenance but it in return might have some problems with security.

11 XMPP Plugin

We have developed a plugin [15] that exchanges references over the XMPP instant messaging network. The XMPP network was chosen because it is open, extendable and secure. It is already possible for two friends to exchange references over XMPP, without the use of a specialized plugin, by simply sending their respective reference to each other as text messages. The plugin in Chapter 12 does this, but works for all major instant messaging networks, including XMPP. This section will only focus on using functionality specific to XMPP in order to exchange references.

In this section the XMPP-specific functionality in the form of extensions will be investigated. This is followed by a description of the libraries used to connect to XMPP servers. Then it is described how the plugin works, including a description of the user interface. Finally there will be an evaluation of the plugin.

11.1 XMPP Extensions

The core functionality of XMPP is quite limited, however many *extensions* exists that adds additional functionality to the protocol. Which extensions that are supported varies between different XMPP servers and clients. It is possible to use the *service discovery* extension to ask a XMPP entity, either a server or a client, which extensions it supports.

11.1.1 PubSub and PEP

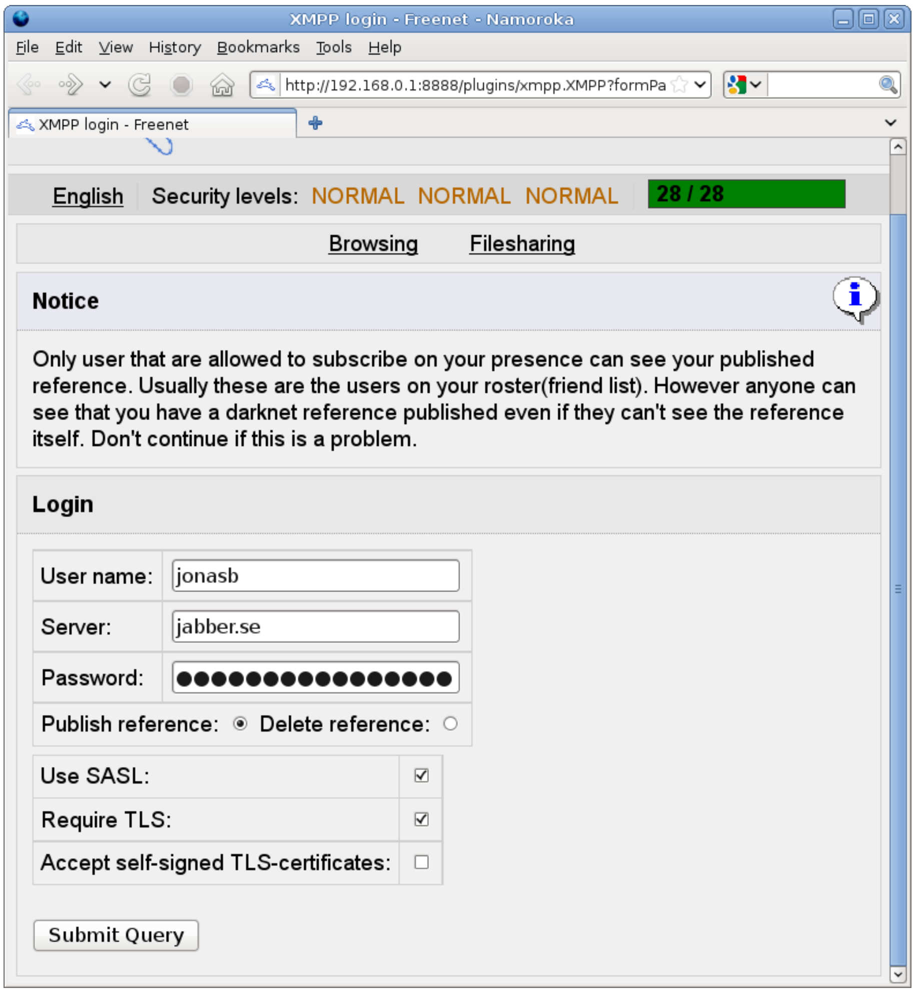
For exchanging references the *PubSub* (Publish/Subscribe) and the *PEP* (Personal Eventing Protocol) extensions were used. PubSub allows users to publish information and for friends to subscribe on it; when the information is changed the friends are informed. It is also possible to retrieve the information published by friends, without subscribing, it is this approach that is used by the plugin. A common usage of PubSub is *content syndication*, a technique which is common on blogs and news sites. Because the PubSub specification is quite complex and not widely implemented, a subset of it was released under the label PEP. With PubSub information is published at a PubSub-service at a location called a *node*. The PubSub-service is often the same as the XMPP-server the user is using. With PEP each account is its own virtual PubSub service.

11.2 Libraries

The plugin uses the Smack library [23] for the XMPP communication, a library developed by Ignite Realtime, the same company that develops the OpenFire XMPP server. Smack has good support for most common XMPP task such as sending text messages, presence information and managing the friend list. However it lacks support for PubSub and its support for PEP is very limited and broken. To get support for PubSub the su-smack library [5] was used, a library developed at the Stockholm University. su-smack was outdated and incomplete and major modifications to it had to be made to function properly.

11.3 Design

The plugin is written in Java and is run as a plugin in FProxy. It has a login page that can be seen in Figure 6. At the login page the user can enter such information as username, password, server and what type of security should be used. The user can also choose if the reference should be published or if already published references should be removed. When the user has logged in the plugin first checks with the service discovery feature if the server allows data to be published, either with PEP or with PubSub. If it is allowed it publishes the reference,which will only be visible to friends. Secondly the plugin adds references for friends.


XMPP login - Freenet - Namoroka
File Edit View History Bookmarks Tools Help
http://192.168.0.1:8888/plugins/xmpp.XMPP?formPa
XMPP login - Freenet
English
Security levels: NORMAL NORMAL NORMAL
28 / 28
Browsing
Filesharing

Notice

Only user that are allowed to subscribe on your presence can see your published reference. Usually these are the users on your roster(friend list). However anyone can see that you have a darknet reference published even if they can't see the reference itself. Don't continue if this is a problem.

Login

User name:	jonasb
Server:	jabber.se
Password:	●●●●●●●●●●●●●●●●

Publish reference: [✓ ●] Delete reference: ○

Use SASL:	✓
Require TLS:	✓
Accept self-signed TLS-certificates:	☐

Submit Query

Figure 6: Login-page for the XMPP plugin. The user enters his username, XMPP-server and password and selects if the reference should be published or deleted. There are also checkboxes for advanced features related to security.

11.4 Evaluation

The main advantage with XMPP is that it is decentralized. The references are scattered across several different servers and no single company or organization controls all of them. A reference can be sent to a server in a secure way over a TLS-connection.

The main problem with using the PubSub functionality is that support is very limited. The largest XMPP-provider Google Talk does not support it at all. In

Appendix C the support for PEP and PubSub are investigated for some popular XMPP server softwares.

There are also issues about privacy. Even if strangers can't see published references, they can see if a reference is published on a specific XMPP account. This is possible because the error messages for the protocol are very verbose. The existence of a reference indicates that the owner of the account is part of a darknet, information that the user may not want to be public.

12 libpurple plugin

It is already possible to send a reference over an instant messaging network, without the use of a specialized plugin, by simply copying it and pasting it in an instant messaging client, however this is quite cumbersome. In an attempt to simplify this, we developed a plugin for libpurple, the library used by the Pidgin and the Finch instant messaging clients. The plugin allows references to be sent over all networks supported by libpurple.

12.1 Design

Initially we had two very different ideas of how the plugin could be implemented. One possibility would be for it to be run as a FProxy plugin that communicated with the Pidgin/Finch process over D-Bus, a bus commonly used for inter-process communication. However, the Java D-Bus library available relied on native code which wouldn't allow the plugin to packaged and distribution in a platform independent manner. Instead we decided to write it as native plugin for libpurple, using the C programming language.

The plugin doesn't use any protocol specific functionality like the plugin in Chapter 11 does, instead it relies only on ordinary text messages which are supported by all instant messaging networks. In order for it to work with users that doesn't have the plugin installed, not only the reference is sent but also a description of how Freenet can be installed. The plugin recognizes these messages and can filter out the reference. To add received references and to get the reference that is sent to friends, the plugin uses the FCP-protocol to communicate with the Freenet node.

In Pidgin, the reference can be sent to a friend simply by right clicking on it and selecting an entry from the menu as can be seen in Figure 7. If the friend also uses the plugin he can accept the reference by a mouse click as can be seen in Figure 8. If the friend doesn't use the plugin, he will see a description on how Freenet can be installed together with the reference.

12.2 Evaluation

The plugin suffers from several major problems. The main problem is about how it should be distributed. Since the plugin is written in C it is not platform independent, hence there is no easy way to bundle it with Freenet.

A problem with the plugin is also that there is no way for a user to see if his friends are part of a darknet, he has either to know in advance which friends are part of the darknet or chat with them to find out. This is not the case for the plugins in the two previous sections.

There is also no big gain in using the plugin. As have already been mentioned the same thing can be accomplished by manually copying and pasting references.

How secure the plugin is depends on which network it is used with. Some networks, such as MSN have by default no support for encryption at all, while other such as XMPP have very good support for security.

13 Further Work

This thesis focuses on how existing social networks on the Internet can be used to grow darknets with the use of specialized plugins. Now we will discuss how future changes to the APIs could affect the plugins and the for developing new plugins. After that we will discuss how darknets could be grown without social networks.

13.1 Changes to APIs

The Facebook plugin uses the Facebook API to exchange references, an API that is under constant development. It is possible that changes will be made to it in the future, which could make improvements to the plugin possible. During the course of the thesis major changes were made to the API which allowed a second improved plugin to be developed. Thus it is not unlikely that improvements could be made to the plugin in the future.

The XMPP plugin uses the PubSub-functionality to exchange references. The specification of PubSub is rather static and it is not likely that new major features will be introduced in the near future. What is likely to happen is that more and more of XMPP servers are getting support for PubSub, which might change the evaluation of the plugin.

Not only the networks we consider could benefit from changes in APIs. During the course of the thesis many different APIs were evaluated in order to find out if they were suitable for growing darknets. Changes are constantly being made to them and it is not unlikely that it will be possible to develop plugins with similar functionality for many more social networks in the future.

13.2 Other ways to grow darknets

There are other potential ways to improve the growth of darknets other than using social networks. Three such ways will be discussed next.

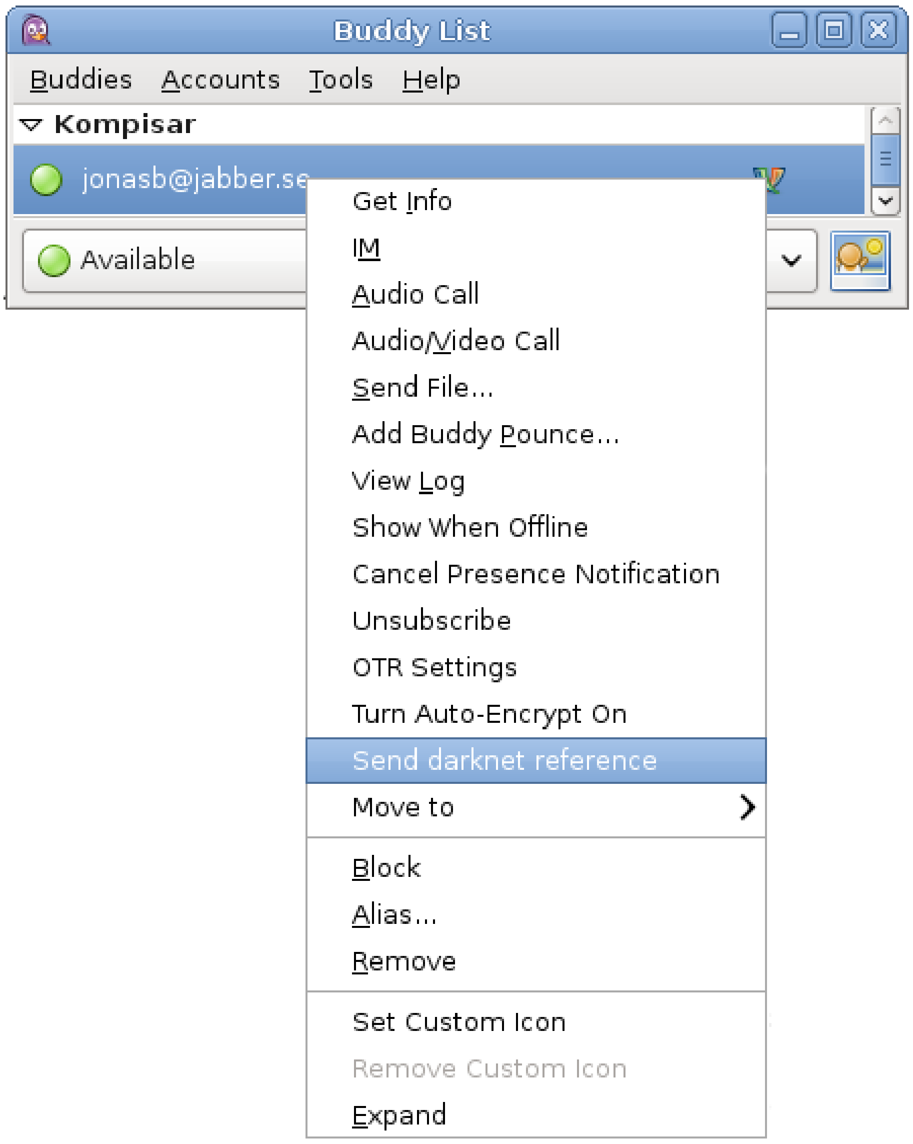


Figure 7: The menu a user gets when right clicking on a friend in Pidgin. The plugin adds an entry for sending a darknet reference. When the entry is pressed the reference is sent automatically.

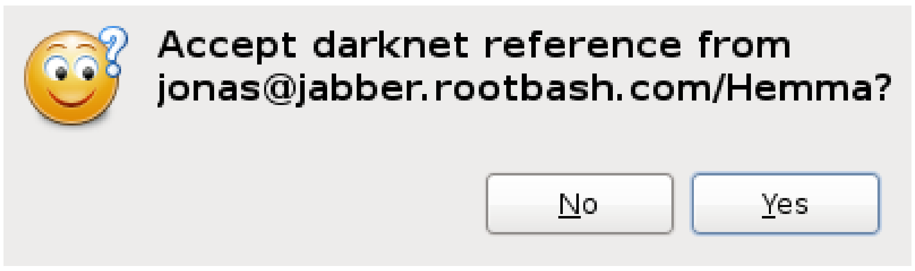

Accept darknet reference from jonas@jabber.rootbash.com/Hemma?

Figure 8: The window that appears when a user of the plugin receives a reference from a friend. If the No-button is pressed, the reference is not added. If the Yes-button is pressed the reference is added automatically.

13.2.1 Using distributed storage

Since Freenet uses a distributed storage, it is possible to store references that are to be exchanged on Freenet itself, without the need of any other network. This can be accomplished if each pair of people that want to exchange references have a common secret. The secret could be used as a key to a location in the network that stores the references. The challenge lies in how the common secrets should be established. It is of course possible to use an existing social network to establish the common secret, but if a social network is to be used, why not simply use it to exchange the reference itself? An other idea is that it might be possible to generate the common secret from some information that both parties already have, such as cryptographic information.

13.2.2 Shorter reference

The Freenet reference is very long as can be seen in Appendix A. This may cause problems if it is sent over e-mail since many e-mail clients wrap lines that are longer than 79 characters. The same may also be true for some instant messaging programs and social networking sites. If the lines of the Freenet references were to be reduced, it would possible to avoid these problems. The reference used for the OneSwarm P2P-network is a lot shorter as can be seen in Appendix B.

13.2.3 Improving friend-to-friend interaction

Making it easier to exchange references is not the only way to get a darknet to grow. Another way is to encourage people to add friends as darknet peers, or even make them persuade friends that are not even using the darknet to start using it. This could be achieved by giving improvements in functionality to people that have darknet friends. It could be possible to add functionality similar to that which is available on most social networking sites, such as profiles and many ways for friends to interact and communicate. A darknet could in theory be used as a substitute for social networking sites.

14 Conclusions

We have investigated if it is possible to grow darknets in a simple and secure manner by using existing social networks. As our darknet of choice we chose Freenet since at the beginning of the work is was the only darknet in wide use.

The darknet-part of Freenet is based around the small world property. In order to illustrate that the property also exists in social networks on the Internet we conducted a case study. We made simulations on the GPG web of trust and the property indeed seemed to exist. The average path length between users in the web of trust was only 5.95 or 5.05, depending on how the graph was constructed. This gave us a conviction that social networks could be used to grow darknets.

Before we started to write our own software that tried to solve the problem, we examined the existing solutions used by OneSwarm. By default it has several ways for exchanging references, but we only considered two of them to be of real interest: Community Servers and GTalk. Both solutions sufferer from the same problem however, they rely on centralized servers, something that goes against the very nature of a darknet.

Instead we wanted to have a solution that exchanged references directly over an existing social network, without the use of any dedicated servers. In order to achieve this we looked into the APIs of all major social networks and looked if they could be of use. For the evaluation to be done in a consistent way we defined four criteria related to security. Of the many networks investigated only three were deemed suitable, the Facebook social networking site, the XMPP instant messaging protocol and the libpurple instant messaging library. We spent a lot of time trying to see if it could work for one of the many networks using the OpenSocial API, but without success. This was unexpected, the open nature of OpenSocial had us believe that it would be easy. For each of the three APIs that were suitable, a plugin was developed.

Of the plugins we consider the one for Facebook to be the one of most importance, since Facebook is the worlds largest social networking site. The plugin is functional but it has some problems related to security. In the Facebook API there is only limited support for setting permissions on published references, this has to be manually by the user, something the user might ignore.

The XMPP plugin works well and allows friends to be added in a simple and automated manner, but it relies on parts of the XMPP protocol that are not widely implemented by XMPP servers. It is also not possible for a user of the plugin to hide the fact that they are using the plugin.

The libpurple-plugin is the plugin of least interest. It simplifies the exchange of references somewhat, but the same functionality can be achieved if the user simply copies and pasting a references and then sending it as a text message to a friend in an instant messaging client like Pidgin. There are also problems with how the plugin should be distributed since the code is written in the platform dependent language C.

All of our plugins suffer from at least some problem, but this is mainly due to limitations of the various APIs. What the plugins do demonstrate is that it is possible to grow darknets in an easy and secure manner with the usage of specialized plugins. At the moment, none of the plugins are suitable for wide usage, but future improvements to the APIs might change this. New and improved APIs might also allow similar plugins to be developed for other social networks.

Something we had to take into consideration when developing and evaluating the plugins was the behaviour of the average user, hence the problem isn't of a strict technical nature. On Facebook for example it is not uncommon for users to have literally hundreds of friends, with most having no in real life relationship with the user. There could be security risks involved if such a user where to use the Facebook plugin, even if there is no technical problems with the plugin itself. In that case the plugin would exchange references between strangers, the same way an opennet works.

References

[1] ICQ Celebrates 100 million Registered Users. `http://www.timewarner.com/corp/newsroom/pr/0,20812,668719,00.html`, 2001.

[2] Myspace: The business of spam 2.0 (exhaustive edition). `http://valleywag.gawker.com/tech/myspace/myspace-the-business-of-spam-20-exhaustive-edition-199924.php`, 2006.

[3] Facebook wins Manx battle for face-book.com. `http://www.theregister.co.uk/2007/10/01/facebook_domain_dispute/`, 2007.

[4] Luke Schierer discusses Pidgin, Open source and life. `http://www.pcworld.idg.com.au/index.php/id;1641709366;pp;1;fp;2;fpid;4`, 2007.

[5] su-smack. `http://static.devel.it.su.se/su-smack/`, 2007.

[6] Alexa Top 500 Global Sites. `http://alexa.com/topsites`, 2009.

[7] Containers. `http://wiki.opensocial.org/index.php?title=Containers`, 2009.

[8] Facebook Statistics. `http://www.facebook.com/press/info.php?statistics`, 2009.

[9] Facebook's New Terms Of Service: "We Can Do Anything We Want With Your Content. Forever. - The Consumerist. `http://consumerist.com/2009/02/facebooks-new-terms-of-service-we-can-do-anything-we-want-with-your-content-fore`
`html`, 2009.

[10] Google Press Center: Press Release. `http://www.google.com/intl/en/press/pressrel/opensocial.html`, 2009.

[11] Google Unveils New Look For Orkut. `http://tech2.in.com/india/news/internet/google-unveils-new-look-for-orkut/93102/0`, 2009.

[12] Opensocial is 1 and reach is 600 million users at 20 sites. http://blogs.sun.com/socialsite/entry/opensocial_is_1, 2009.

[13] Share your favorite personal Windows Live Messenger story with the world! http://messengersays.spaces.live.com/Blog/cns!5B410F7FD930829E!73591.entry, 2009.

[14] Source code for Facebook plugin. http://github.com/ljb/freenet-facebook/, 2009.

[15] Source code for XMPP plugin. http://github.com/ljb/freenet-xmpp/, 2009.

[16] Using Batching API. http://wiki.developers.facebook.com/index.php/Using_batching_API, 2009.

[17] XSF Press Release. http://xmpp.org/xsf/press/2003-09-22.shtml, 2009.

[18] API - Facebook Developer Wiki. http://wiki.developers.facebook.com/index.php/API, 2010.

[19] facebook-java-api. http://code.google.com/p/facebook-java-api/, 2010.

[20] Facebook No Longer The Second Largest Social Network. http://techcrunch.com/2008/06/12/facebook-no-longer-the-second-largest-social-network/, 2010.

[21] Freenet-Cards. http://freenetcard.draketo.de/, 2010.

[22] HttpComponents. http://hc.apache.org/, 2010.

[23] Ignite Realtime: Smack API. http://www.igniterealtime.org/projects/smack/, 2010.

[24] MSN Messenger encryption and security software. http://www.secway.fr/us/products/simplite_msn/home.php, 2010.

[25] Orkut member statistics. http://www.orkut.com/MembersAll.aspx, 2010.

[26] People Against the new Terms of Service (TOS). http://www.facebook.com/group.php?gid=77069107432, 2010.

[27] pidgin-facebookchat. http://code.google.com/p/pidgin-facebookchat/, 2010.

[28] Pidgin, the universal chat client. http://www.pidgin.im/, 2010.

[29] The Freenet Project - /faq. http://freenetproject.org/faq.html#blocked, 2010.

[30] The Freenet Project - /philosophy. http://freenetproject.org/philosophy.html, 2010.

[31] The GNU Privacy Guard. http://www.gnupg.org, 2010.

[32] Wotsap. http://www.lysator.liu.se/~jc/wotsap/, 2010.

[33] Ian Clarke, Oscar Sandberg, Matthew Toseland, and Vilhelm Verendel. Private Communication Through a Network of Trusted Connections: The Dark Freenet. 2010.

[34] Thomas H Cormen, Charles E Leiserson, and Ronald L Rivest. *Introduction to Algorithms*. MIT Press and McGraw-Hill, 1990.

[35] Peter Sheridan Dodds, Roby Muhamad, and Duncan J. Watts. An Experimental Study of Search in Global Social Networks. 2006.

[36] Tomas Isdal, Michael Piatek, Arvind Krishnamurthy, and Thomas Anderson. Friend-to-friend data sharing with OneSwarm. Technical report, University of Washington, 2009.

[37] Jon Kleinberg. Navigation in a small world. *Nature 406*, 2000.

[38] Jure Leskovec and Eric Horvitz. Planetary-scale views on an instant-messaging network. 2007.

[39] Oskar Sandberg. *The Structure and Dynamics of Navigable Networks*. PhD thesis, Gothenburg University and Chalmers University of Technology, 2007.

[40] Flora Rheta Schreiber. *Sybil*. 1973.

Appendix A Freenet Reference[4]

```
opennet=false
identity=3YhCVM9wyh-tLRfqtiwAke~sftxBzqDPKol5wdO0SJ0
myName=Jonas@Chalmers
lastGoodVersion=Fred,0.7,1.0,1198
sig=529684620778051277c5d8797ba3cccf37a043c2a124fe4ab9a6c40c07e3ed240,52b310ad78
    9913d7832d67b9d49fc5f84104375dee7f30fd45bdc8ecfe980555
version=Fred,0.7,1.0,1205
dsaPubKey.y=GuisG6ku8IBDAuz8BjRA2ldpqqIpUouhRLpco9cAFvIJocsrnvZwvo3oUf2FHnpJYyA
ng lines wrapped and indentation added.
    92lBBhpmzF-CC23HRDlPNfihAVkQQmpcRsmT9l4X3ryg5krvjhYtwgnbhCHEILk2TQJxtBlf5Li
    6V9Dmv1cwHb77pO~GiWSeMkGLuPPOPwR0Vu2M9iGO3W-1SYwg6iWCH2rkEvTyTl2kBBDyfO-ZvY
    Or2mhKga4RxsKr9486bL2ZeUqvFjgFckd3h1tcC9q9gaU3Fe1EkwgAWBovZnPcJIadwfBeNBnJ3
    DPq9RqR~bKE5~Y1R2YarLP-i1y2Nhj5qYzYLHj8-svieG0onPA
physical.udp=fe80:0:0:0:20b:dbff:feba:e2ed%2:15161;129.16.22.10:15161;
    0:0:0:0:0:0:0:1%1:15161;127.0.0.1:15161
dsaGroup.g=UaRatnDByf0QvTlaaAXTMzn1Z15LDTXe-J~gOqXCv0zpz83CVngSkb--bVRuZ9R65OFg
    ~ATKcuw8VJJwn1~A9p5jRt2NPj2EM7bu72O85-mFdBhcav8WHJtTbXb4cxNzZaQkbPQUv~gEnuE
    eMTc80KZVjilQ7wlTIM6GIY~ZJVHMKSIkEU87YBRtIt1R~BJcnaDAKBJv~oXv1PS-6iwQRFMynM
    EmipfpqDXBTkqaQ8ahiGWA41rY8d4jDhrzIgjvkzfxkkcCpFFOldwW8w8MEecUoRLuhKnY1sm8n
    nTjNlYLtc1Okeq-ba0mvwygSAf4wxovwY6n1Fuqt8yZe1PDVg
dsaGroup.q=ALFDNoq81R9Y1kQNVBc5kzmk0VvvCWosXY5t9E9S1tN5
dsaGroup.p=AIYIrE9VNhM38qPjirGGT-PJjWZBHY0q-JxSYyDFQfZQeOhrx4SUpdc~SppnWD~UHymT
    7WyX28eV3YjwkVyc~--H5Tc83hPjx8qQc7kQbrMb~CJy7QBX~YSocKGfioO-pwfRZEDDguYtOJB
    HPqeenVDErGsfHTCxDDKgL2hYM8Ynj8Kes0OcUzOIVhShFSGbOAjJKjeg82XNXmG1hhdh2tnv8M
    4jJQ9ViEj425Mrh6O9jXovfPmcdYIr3C~3waHXjQvPgUiK4N5Saf~FOri48fK-PmwFZFc-YSgI9
    o2-70nVybSnBXlM96QkzU6x4CYFUuZ7-B~je0ofeLdX7xhehuk
ark.pubURI=SSK@CtfkAHX9E4v5QO7AmViY9yFljSP5z5R8Vs97QcBvYXU,SF4Hlbi3kj4MAdqbofsj
    Pn8U7lcLDC5i0ieX07K9qVE,AQACAAE/ark
ark.number=1
auth.negTypes=2;4
End
```

Appendix B OneSwarm Reference

```
MIGfMA0GCSqGSIb3DQEBAQUAA4GNADCBiQKBgQC8O6Seh
dS7K25cWrIIDNobk9xw1OtVqJFTHy1zP5b8c6wD10qS2L
BeCyKhUIQpl9vvM5efIht0PXkDDhwp1wQ2BmNh7wKNxym
JJbB75RfZrXXzthHNoIwokSFfasWNQwrN/kMgt4T/nsaL
9oD/MynQE9Tlko+MlsFPxL5C196EJQIDAQAB
```

[4]Long lines wrapped and indentation added

Appendix C Support for PubSUB and PEP Extensions to XMPP

Support for PubSub and PEP extensions to XMPP is limited, as can be seen in Table 1.

Name	PubSub support	PEP support
Google Talk	No	No
jabberd2	No	In development
tigase	Yes	Yes
psyced	No	No
gtalk	No	No
ejabberd	Yes	Yes
OpenFire	Yes	Yes

Table 1: Support of PEP and PubSub for some popular XMPP servers.

Even if a server has support for PubSub or PEP, it doesn't mean that it supports all the features. The specifications of PubSub is very complicated. Even between different versions of the same server there can be some differences in level of support as Table 2 shows.

Feature	ejabberd 2.0.3	ejabberd 2.0.4	OpenFire 3.6.3
pep	Yes	Yes	Yes
pubsub	Yes	Yes	Yes
pubsub#access-authorize	No	No	No
pubsub#access-open	Yes	Yes	No
pubsub#access-presence	Yes	Yes	No
pubsub#access-roster	No	No	No
pubsub#access-whitelist	No	Yes	No
pubsub#auto-create	Yes	Yes	No
pubsub#auto-subscribe	Yes	Yes	No
pubsub#collections	Yes	Yes	Yes
pubsub#config-node	Yes	Yes	Yes
pubsub#create-and-configure	Yes	Yes	Yes
pubsub#create-nodes	Yes	Yes	Yes
pubsub#delete-any	No	No	No
pubsub#delete-nodes	Yes	Yes	Yes
pubsub#filtered-notifications	Yes	Yes	No
pubsub#get-pending	No	No	Yes
pubsub#instant-nodes	Yes	Yes	Yes
pubsub#item-ids	Yes	Yes	Yes
pubsub#last-published	Yes	Yes	No
pubsub#leased-subscription	No	No	No
pubsub#manage-subscription	No	No	No
pubsub#member-affiliation	No	Yes	No
pubsub#meta-data	No	No	Yes
pubsub#modify-affiliations	Yes	Yes	Yes
pubsub#multi-collection	No	No	No
pubsub#multi-subscribe	No	No	Yes
pubsub#outcast-affiliation	Yes	Yes	Yes
pubsub#persistent-items	Yes	Yes	Yes
pubsub#presence-notifications	Yes	Yes	Yes
pubsub#presence-subscribe	Yes	Yes	No
pubsub#publish	Yes	Yes	Yes
pubsub#publish-options	No	No	No
pubsub#publisher-affiliation	Yes	Yes	Yes
pubsub#purge-nodes	Yes	Yes	Yes
pubsub#retract-items	Yes	Yes	Yes
pubsub#retrieve-affiliations	Yes	Yes	Yes
pubsub#retrieve-default	Yes	Yes	Yes
pubsub#retrieve-items	Yes	Yes	Yes
pubsub#retrieve-subscriptions	Yes	Yes	Yes
pubsub#subscribe	Yes	Yes	Yes
pubsub#subscription-options	No	No	Yes
pubsub#subscription-notifications	Yes	Yes	No

Table 2: Detailed comparison for support of PEP and PubSub for three different XMPP servers.