Further reducing the anonymity set of web servers hidden within the I2P network

Iwan Hoogendoorn - iwan.hoogendoorn@os3.nl
Tarik El Yassem - tarik.elyassem@os3.nl
Joris Soeurt - joris.soeurt@os3.nl

December 23, 2011

Abstract

In this SSN project we have tried to reproduce the results of the paper "Darknets and hidden servers: Identifying the true IP/network identity of I2P service hosts" by Adrian Crenshaw (2010) as a baseline and subsequently tried to improve on these results. In this paper he describes methods of linking eepSites (hidden websites within the I2P darknet) to IP addresses. His research assumes application (configuration) errors, and is mainly based on header comparison of webservers that are both available on I2P as IP. He also uses different techniques to further reduce the anonymity set of these eepSites. We've tried to reproduce these results and subsequently improve on them by using new techniques, mainly based on metacrawling of eepSites and enlarging the sets of eepSites and IP addresses. Although we have gained much knowledge about the architecture and design of the I2P overlay network, unfortunately we didn't succeed on improving the accuracy of identify these eepSites. The main reason for this is that the tactic Crenshaw uses turned out to be not that relevant anymore; almost all sites we could find online only showed generic headers instead of very specific package and version information. Also, the crawling of metadata didn't deliver any valuable information. Although we did find hashes of eepSites that were not publicized in the main public address books, we choose not to continue with these eepSites because of the (probably illegal) content presented on the index page of one of these sites.

Contents

1 Introduction

Several privacy enhancing overlay networks exist. Beside the largest and most popular network Tor, the I2P network is also relatively large. The major design difference with Tor is that I2P mainly focuses on hosting services within the network instead of creating an anonymous gateway to regular Internet services. Adrian Crenshaw has written a paper[1] and proof of concept of how to link these hidden anonymous services to real world (non anonymous) IP addresses. Although the paper seemed clear, we thought there was room for improvement of various aspects. Important to this is that this paper originates from 2010. Because of the increased interest in anonymization networks, it would also be interesting to see if we could still achieve the similar results.

Our main research question is therefore:

"Is it possible to increase the percentage of eepSites (I2P hosted websites) that can be linked to real world IP addresses by improving on, and using techniques yet unused by Crenshaw?"

The sub questions are defined as:

- What effect does enlarging the set or known eepSites to the percentage of identified hosts?
- Is crawling metadata an effective method of reducing the anonymity of eepSites?
- What effect does enlarging the set of known IP addresses of I2P participants have to the number of linked eepSites?

1.1 Related Research

We found the following research papers to be relevant to our research:

Privacy-Implications of Performance-Based Peer Selection by Onion-Routers: A Real-World Case Study using I2P[3]
Author: Michael Herrmann of the Technical University of Munich
Description: This paper presents an attack on the I2P network, with the goal of determining the identity of peers that are anonymously hosting HTTP (eepSite) services in the network. The attack is basically finding out the peers of a certain target, attacking these peers through Denial of Service attacks such that the peers fall away and are replaced. The attacker manipulates this process with the goal of introducing systems under his control as peers of the target. This allows the attacker to learn the targets identity by making measurements.

Monitoring the I2P network[2]
Authors: Juan Pablo Timpanaro, Isabelle Chrisment, Olivier Festor from the INRIA research centre
Description: Timpanaro, Chrisment and Festor have performed a one week long experiment to monitor the I2P network and have found that webservers hosted within I2P remain at a specific address longer than servers that offer bittorrent services.

Darknets and hidden servers: Identifying the true IP/network identity of I2P service hosts[1]
Authors: Adrian Crenshaw
Description: Crenshaw's paper explains various techniques that can be used to link an eepSite address to an IP address that participates in the I2P network. This paper discussed many ideas we formulated during the pre-research stage. Crenshaw demonstrated in his paper that a small subset of eepSites could be linked

to IP addresses of nodes participating in the I2P network. The paper makes a few suggestions on other techniques that can be applied to increase the number of links that can be found. Since this paper forms a basis of our research, we suggest the reader to also read this paper.

1.2 Approach

- Try to reproduce Crenshaw's results using the same methods he decribes to establish a baseline for improving.
- Select methods to improve on baseline results.
- Use selected methods to improve on baseline results.

2 Background on I2P

2.1 Background on Anonymization Networks

Events over the past years emphasized the importance of having a anonymous, uncensored and private means of communication. Think of events such as the Arab spring, three-strikes law (France) and deep packet inspection of Dutch telecom operators.
The reasons why one wants to operate anonymous on the internet can be categorized as follows[4]:

- Support of free speech.
- Censorship resistant communication.
- Preserve privacy.
- Distribute materials.

To have a good anonymity model not only the content of the messages but also both initiator and receiver's identity should be hidden. The challenge inherent to this issue is finding a way of communicating without explicitly referring to both sender as receiver.
The only way this can be fully achieved is a distributed model such as in overlay networks like I2P or Tor where no central authority is responsible and communicating parties can refer to each other in anonymized way.

2.2 What is I2P?

I2P is an anonymizing overlay network. The I2P application itself is implemented as a message orientated middleware. Applications can use this middleware to communicate to other nodes, identified by a cryptographic ID. If implemented correctly the system guarantees no link can be made between a host's I2P address and IP address. The relation between the identity (IP address) and I2P ID (cryptographic identifier) of a node cannot be discovered by participating nodes. All aspects of the I2P network are open source and available without cost, as this should both assure the people using it that the software does what it claims, as well as enable others to contribute and improve upon it to defeat aggressive attempts to stifle free speech. There are several application that can interface directly with the I2P network. These applications include blogging, file sharing, email and instant messaging.

2.3 Concepts

To understand the working of the I2P protocol, it is important to understand several concepts which are explained below.

2.3.1 The netDb

I2P uses a distributed database called the netDb. This database is maintained only on the superpeers, (named floodfill peers in I2P) and contains all needed information to make the network operate.

Two types of information are stored in this database on these floodfill peers:

- **routerInfo** entities, which contain all the information needed to contact a particular router. (Routers in I2P have the same concept as regular routers, they are intermediate nodes when delivering traffic between two nodes).
- **leaseSets**, which represent a particular destination. The leaseSet contains all information needed to contact this destination (inbound gateway to reach that particular destination, time when the tunnel expires, pair of public keys to reach the destination).[5]

2.3.2 Floodfill peers

These floodfill peers are automatically selected between all nodes. If the number of floodfill nodes drop below a certain percentage, peers that are fast enough and have enough bandwidth are promoted to floodfill peer.

2.3.3 Replication

Other regular I2P routers (no floodfill) store their information by sending store requests to floodfill peers which is then subsequently spread between the superpeers by use of the **Kademlia** algorithm[9]. When a node needs information to contact a specific endpoint it sends a lookup request to a floodfill peer.

2.3.4 Bootstrapping

As with every fully distributed protocol, bootstrapping is an issue. For I2P, this problem is handled by hardcoded links to regular internet sites which contain the routerInfo sets of some arbitrary I2P nodes. Once a new I2P peer has managed to obtain such a routerInfo record of an arbitrary node, it can query that node for further network information, such as the address of a supernode.[5]

2.3.5 Tunnels

I2P node use in- and outbound tunnels. These tunnels are a directed path between two nodes. Traffic from source to destination is forwarded from an outbound tunnel of the source, to an inbound tunnel of the destination. Traffic in the same conversation that travels from the destination back to the source, uses the same concept, but follows another path.[7]

2.4 Communicating with other hosts

When a node comes online, it creates a couple of in- and outbound tunnels using semi-random peers. It then advertises its own inbound gateway (endpoint of inbound tunnel) to a floodfill peer. When the node wants to contact another node, it asks the floodfill peer for the inbound gateway of the destination node. It then sends the message into the outbound tunnel, tagged with the address of the inbound gateway of the destination node. The intermediate nodes in the outbound tunnel of the source and inbound tunnel of the destination make sure the message gets delivered at the correct node.[8]

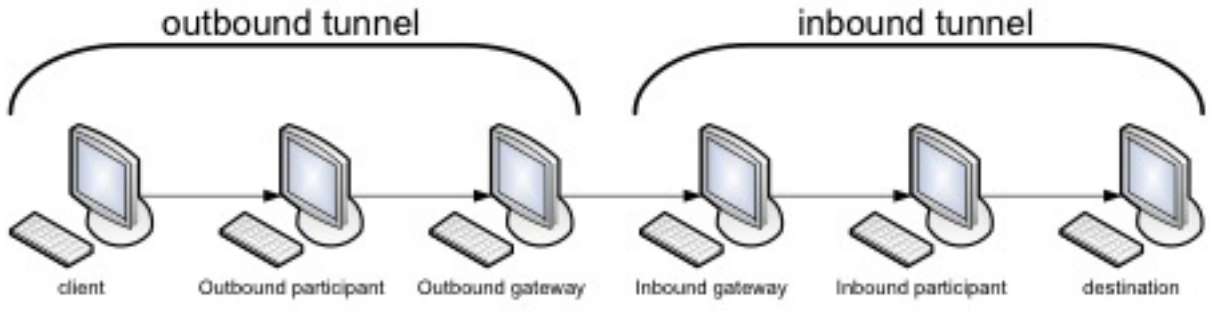

Figure 1: inbound and outbound tunneling: source www.i2p2.de

A tunnel length can be between of 0 and 7 nodes, although recent studies have showed that a length of more than 3 nodes does not provide additional protection.[7]

2.5 I2P addressing

There is no central naming authority in I2P. Nodes are identified by a unique 516-bit hash, and local host files are used to map human readable names to these hashes. These name-to-hash mappings can be shared in address books, based on a web-of-trust. Different people can have local address book entries for Alice

which refers to different addresses. New hosts can be added by importing published address books or by adding hosts to the hosts file[10].

I2P hosts can also have a .i2p hostname which must conform to the naming rules specified in RFC2396[12]. If an application wishes to access a destination by name, the router does a local lookup to resolve that name. The client application does a linear search through three local files to look up host names and convert them to the 516-byte destination key. The local files are[5]:

- privatehosts.txt
- userhosts.txt
- hosts.txt

All destinations in I2P are 516-byte keys. The keys consist of a 256-byte public key plus a 128-byte signing key plus a null certificate, which in Base64 representation is 516 bytes. The certificate is null because at this moment certificates are not used.

2.5.1 Base32 hostnames

I2P also supports Base32 hostname. I2P uses 52 characters (256 bits) to represent the full SHA-256 hash. The form is 52 chars.b32.i2p. Base32 is implemented in the HostsTxt naming service, which queries the router over I2CP to lookup, the LeaseSet to get the full destination. Base32 lookups will only be successful when the destination is up and publishing a LeaseSet. Base32 addresses can be used in most places where hostnames or full destinations are used, however they may fail if the name does not immediately resolve[11].

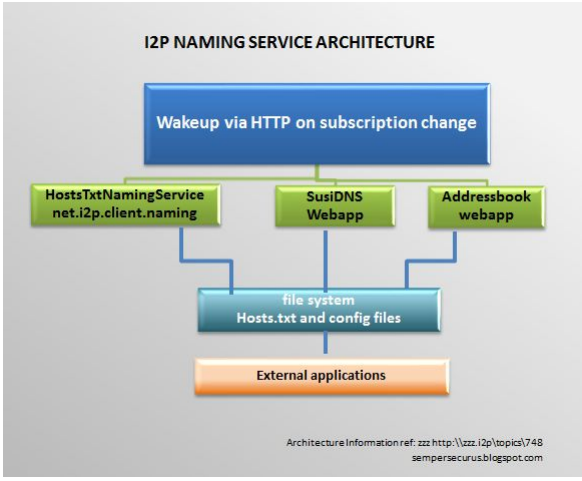

Figure 2: i2p naming service architecture: source http://sempersecurus.blogspot.com

2.6 I2P Crypto

I2P uses different cryptographic algorithms in a layered way for the various types of communication. I2P uses the following communication levels:

- Inter router communication.
- Tunnel messages.
- Garlic messages.

2.6.1 Inter router communication

This is the lowest level of communication between routers. Routers select an ephemeral session key through a 2048 bit Diffie-Hellman exchange. Each router authenticates to the next router by using it's DSA key. Each packet sent between routers is encrypted with AES256/CBC with an explicit IV and MAC (HMAC-MD5-128). A hash of each packet is used for local integrity checking. This layer is illustrated in the picture below, it applies to communication from A to B, B to C, C to D, D to E, E to F, F to G and G to H.

2.6.2 Tunnel messages

Tunnel messages are exchanged on the **middle layer** of the I2P stack. Tunnel messages passed over the transporting routers use their own AES256/CBC encryption with an explicit IV. Messages are verified at the tunnel endpoint with an additional SHA256 hash. The tunnel messages of this middle layer are represented in the picture below. It applies to communication from A to D and from E to H.

2.6.3 Garlic messages

The top layer, the full path combining tunnels, uses another encryption scheme. Messages are passed along inside garlic messages, which are encrypted with ElGamal/AES+SessionTags. Garlic messages are somewhat similar to onion messages as used by Tor. However, in garlic messaging the contents of a single message contains multiple cloves. Cloves are data messages with instructions for delivery. The sender's router put the data message into a garlic, encrypt that garlic to the 2048bit ElGamal public key published in the recipient's leaseSet, and forward it through the appropriate tunnels. The delivery instructions attached to each clove includes the ability to request that the clove be forwarded locally, to a remote router, or to a remote tunnel on a remote router. The full path tunnels of this top layer are illustrated in the picture below as communication between A and H.

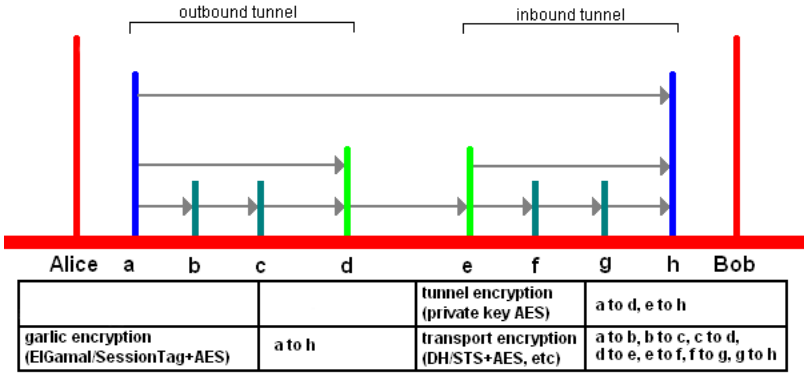

Figure 3: encryption layers in i2p: source http://www.i2p2.de

3 Reproduction of Crenshaw's results

3.1 Introduction

In order to see if we can optimize the results of identifying eepSites, we first needed to reproduce Crenshaw's research in order to establish a baseline. In order to reduce the anonymity set of eepSites, Crenshaw's uses a number of techniques. He also proposes some other techniques without actually using them.

Table 1: Crenshaws paper describes the following research steps

Method	**Scripts**
Banner grabs of both eepSites and banner grabs of web-servers that are hosted on IP addresses which participate in the I2P overlay network. He then compares these if two servers give back the same very uncommon banner. This is a indication it might be the same server.	**I2PMassGrabber-Headers.py** **real-IP-banner.py**
Testing I2P virtual host names on the public facing IP of I2P nodes	**virtual-server-test.py**
Reverse DNS to find out more information concerning the IPs of the I2P nodes	none
Geo-IP lookups to find out more information concerning the IPs of the I2P nodes	none
TCP/IP stack OS fingerprinting	standard tools
Compare the clock of the remote I2P site, and suspected IP hosts on the public Internet, to our own systems clock.	**time-stamp-server.py**

The methods he proposes but doesn't use are:

- Scanning sites for metadata.
- Command injection attacks.
- Webbugs to attempt to de-anonymize eepSite administrators or users.

We chose not to reproduce the **TCP/IP stack OS fingerprinting** because this is only possible over IP links and not I2P links and therefore the results are rather insignificant for the research. Also, we did not extend the research to **command injection attacks** and **webbugs** because they boil down to hacking. A law enforcement or intelligence agency might use these techniques in some situations however.

We split up the remaining steps to take, as follows:

- Add (same) host files as Crenshaw.
- Scrape host file for I2P addresses, and grab HTTP banners.
- Scrape netDb for IP addresses, and grab HTTP banners.
- Reverse DNS to gather more information.
- Enumerate virtual server names.
- Compare the clock offset of I2P sites and IP hosts.
- Geo-IP lookup of IP addresses to gather more information.
- Link eepSites to IP addresses.

3.2 Test environment

In order to reproduce the research we used hardware that was provided by the SNE Master. The main reason for this choice (in contrast to using our home systems) was the large bandwidth availability. We used remote connections (RDP & SSH) to connect from our home locations.

In the table below we describe the configuration of the machines on which the results of these research are based. Other virtual machines were deployed on the same hardware too, but only for pre-research testing (getting acquainted with the software) Although VMware is proprietary software, we choose this product as our virtualization layer because of our experience with this product in combination the time constraints which were subject to.

The reproduction of the results as a baseline was done on a Windows based system because this is the same as Crenshaw did. We also tried to do this on an Ubuntu system, but it turned out the Python scripts had to be adjusted too much to be plausible within this project.

Table 2: Overview of systems used in this project

	ESX Host	**Machine 1**	**Machine 2**
Purpose	Hosting our test environment	Reproduction of research	Improving on results
IP address	145.100.104.53	145.100.106.7	145.100.106.13
Operating system	VMware vSphere 5	Windows 2003 SP2	Ubuntu 10.04.3 LTS Desktop
Hardware	Dell R210	-	-
Internal memory	8 GB	1 GB	1 GB
Internet connection	1 GB shared SURFnet	1 GB shared SURFnet	1 GB shared SURFnet
IP2 Client	-	0.8.11-0	0.8.11-0
Zed Attack Proxy (ZAP)	-	1.3.4	-
Java (JRE)	-	1.6.0_29	1.6.0_20
Python	-	2.7	2.6
geoiplookup	-	-	1.4.6.dfsg-17
geoip database	-	-	Dated: 2011-12-06

We also installed the scripts used by Crenshaw to reproduce the research that is described in the following section. Note that we had to alter these scripts to make it work specifically in our environment.

3.3 Verifying the installation

After installation of all software components, we verified I2P was correctly installed and operating using the user interface. It indicated the number of acting peers as 29 / 83, and explicitly displayed that I2P was working correctly. We verified this by configuring our browser to use localhost:4444 as proxy server and visited the site `http://hq.postman.i2p`.

3.4 Reproducing results as baseline

3.4.1 Step 1: Add (same) host files

To start with the same environment as Crenshaw, we added the I2P host files that Crenshaw mentions in his paper. These host files map user-friendly names (like host.i2p) to I2P hashes needed to connect to an eepSite. These files are maintained by I2P participants in a decentralized way.

The files that we used in order to dynamically retrieve the host files:

```
http://www.i2p2.i2p/hosts.txt
http://i2host.i2p/cgi-bin/i2hostetag
http://stats.i2p/cgi-bin/newhosts.txt
http://tino.i2p/hosts.txt
```

After adding these host files, our local address book contained 374 entries (on Friday 2 December at 10:25 pm GMT+1). Because an I2P connection sometimes times out, not all address books might had been downloaded yet. Unfortunately there is no way of verifying this in the user interface of the client. To make sure we had all entries, we left the setup in this state for a while to see if any more entries would follow.

At 10:34 we had 585 entries, so we decided to wait some more.
At 11:00 the entries remained at 585.

Crenshaw had 1538 host names in his address book on 27-10-2010 at approximately 01:00 pm EST.

To make sure we had all the entries available in the host files, we used a workaround. We downloaded the host files manually, concatenated them, made them unique, and then imported the file manually in the I2P client (by overwriting the current file on disk and subsequently restarted the client).

Using this method we found out that tino.i2p was offline, so we were unable to use the entries defined in this file. Using this method of manually downloading the other three address books resulted in 2,113 unique entries.
This difference of about a 500 entries can probably be explained by the growth of the I2P network since Crenshaw research.

3.4.2 Step 2: Scrape host file for I2P addresses, and grab HTTP banners

Crenshaw's next step was setting up the ZED attack proxy and running **I2PMassGrabber-headers.py** on his host file.

Crenshaw reported the following results:

- A scan with 100 concurrent threads found 104 active eepSites in 798.585 seconds.
- A scan using 10 threads found 112 in 5934.425 seconds.

First, we had to modify the location of the hosts.txt file in the **I2PMassGrabber-headers.py** script. This script enumerates our local address book and tries to connect to every entry, and if online, grabs the banner. Next, we had chain the Zed Attack Proxy to the I2P proxy of our local client (localhost:4444) and configure the script to use the Zed Attack Proxy (localhost:8080)

Note that I2P client offers a HTTP(S) proxy for the host it runs on, on port 4444 which you should use the tunnel traffic into the I2P network.

Next we ran the script. It threw lots of errors and there were only few hits. We changed the threads to 10 but this did not make a difference.

An example of the errors:

```
(<class 'urllib2.HTTPError'>, HTTPError(), <traceback object at 0x00BDEEB8>)
(<class 'urllib2.HTTPError'>, HTTPError(), <traceback object at 0x00BDE670>)
(<class 'urllib2.HTTPError'>, HTTPError(), <traceback object at 0x00BE2328>)
(<class 'urllib2.HTTPError'>, HTTPError(), <traceback object at 0x00BE2670>)
(<class 'urllib2.HTTPError'>, HTTPError(), <traceback object at 0x00BE2A30>)
"docs.i2p2.i2p","200","blank"
```

The index page of found and scraped I2P sites is saved to a subdirectory of name **I2PPageScrapes-date-.** We ran the script a few times to see if it would yield different results, which it did. We observed varying results in a short amount of time. We observed a total number of 28 unique, active eepSites over multiple scans.

In contrast to Crenshaw's scan, in our scan most of the hits returned a "blank" server string, and many returned a simple string such as "Apache" with no version number.

Example of the output:

```
"ugha.i2p","200","blank"
"forum.i2p","200","blank"
"freshcoffee.i2p","200","Apache"
Scan started at 2011-12-03-00-58-44.944000 with 10 threads
Scan ended 2011-12-03-01-10-42.412000
```

Some observations we made while running the script multiple times were:

- The script does not set a common user-agent string, a Python library name is shown. The servers could block this.
- The errors might be overcome, or might give more interesting results by handling the "HTTPError" errors.
- It was unclear in what matter Crenshaw used the ZED attack proxy output. No scripts seem to use this data.
- Running the scripts over a longer period of time will result in more data.
- Although we easily found as many as 2113 host entries of eepSites.

3.4.3 Step 3: scrape netDb for IP addresses, and grab HTTP banners

In order to generate a list with the IP addresses we used the script **"dump-and-sort-i2p-router-ips.py"**. This script uses the local netDb cache to obtain IP addresses of I2P peers.

I2P's netDb is a specialized distributed database, containing just two types of data - router contact information (RouterInfo's) and destination contact information (LeaseSets). Each piece of data is signed by the appropriate party and verified by anyone who uses or stores it. In addition, the data has liveliness information within it, allowing irrelevant entries to be dropped, newer entries to replace older ones, and protection against certain classes of attack. The script retrieves all IPv4 addresses with the use of regular expressions and pipes that into a single file.

Crenshaw ran the script on November 9th 2010 which reported 1099 IP addresses.
We ran the script on December 3rd 2011 and the script reported 395 IP addresses.
We ran the script on December 13th again and also found 395 IP addresses.

This significantly lower number can be explained by the following:
Nodes get into the netDb as a result of having had a connection to form the 'cloves', the tunnels. Since we had less address book entries, we made less scans which results in less traffic and less tunnels, this influences the IP's that can be found. Unfortunately, we did not have enough time to test this theory.

Next we used the **real-IP-banner.py** script to grab banners of these IP peers, which offer an webserver through their regular IP endpoint. This script takes the output of **dump-and-sort-i2p-router-ips.py**

Crenshaw found 1538 IP addresses of which 172 (which is about 11,18 percent) were running a webserver that returned status code 200. It is unclear if Crenshaw only found 172 webservers which all returned a status code of 200, or if he found more webservers of which 172 gave a status code of 200.

We found 73 sites on 393 IP addresses, of which 57 gave a status code of 200. When we ran the script we found 73 websites being hosted on these 395 IP's, which is about 19.7 percent.

3.4.4 Step 4: Reverse DNS

Crenshaws paper suggests using reverse DNS on the IP addresses. However no scripts were available to perform reverse DNS and use this data. Hence we created a script that takes the output file of the **dump-and-sort-i2p-router-ips.py** script and performs a reverse DNS lookup. The script prints and stores the results to a file. This output can be used in step 4 to guess virtual hostname.

3.4.5 Step 5: Enumerate virtual server names

The **virtual-server-test.py** script required a CSV with IP addresses and I2P hostnames. Because we could not make assumptions on what I2P hostname might be used as a virtual server name, we had to create the Cartesian product of IP addresses and I2P hostnames. We created a small script that generated the Cartesian product based on the output of the **I2PMassGrabber-headers.py** and **dump-and-sort-i2p-router-ips.py** scripts.

We used the output of our script as input for **virtual-server-test.py**, unfortunately many IP's seemed to be down and no useful results were returned.

We could have tested that the script was working by hosting an eepsite and website ourselves and register an I2P address. We could also try to run the script at a few other moments. Unfortunately, due to time constraints, for now we must conclude that this method is not effective anymore.

We have not used the output of step four, the hostnames gathered from reverse DNS, because we wanted to reproduce Crenshaws research and compare results. The paper only mentions reverse DNS as a way to gather more information, and does not provide information on if and how this was done and what the results were.

3.4.6 Step 6: Compare the clock of I2P sites and IP hosts

One of the techniques used by Crenshaw was getting timestamps from the HTTP header of eepSites and IP facing webservers and comparing them to the local clock. The time difference is then used as a characteristic of a specific system and may help in identification. This technique is performed by the **time-stamp-server.py** script. We ran this script multiple times on different hours to check if the timeskew was consistent and if we could find any lines that suggested a link between an IP address and an eepsite. Unfortunately, we did not find any occurrences from which we could reasonably conclude that there was a link. The output of **time-stamp-server.py** is a file with timestamps; there are no other scripts to interpret these.

```
 1.434,0.266,"Tue, 13 Dec 2011 00:16:56 GMT",76.250.48.108,lighttpd/1.4.28
-1217.551,0.39,"Mon, 12 Dec 2011 23:56:37 GMT",75.145.125.61,Apache/2.2.21 (Unix) mod_ssl/2.2.21 OpenSSL/1.0.0d PHP/5.3.6
-63459332219.6,0.156,"Tue, 1 Jan 0001 00:00:00 GMT",77.37.172.127,blank
-139.582,0.109,"Tue, 13 Dec 2011 00:14:41 GMT",77.64.153.140,Apache/2.2.17 (Win32) mod_ssl/2.2.17 OpenSSL/0.9.8o PHP/5.3.4 mod_perl/2.0.4 Perl/v5.10.1
11.418,0.531,"Tue, 13 Dec 2011 00:17:27 GMT",78.231.24.103,Apache/2.2.14 (Ubuntu)
1.403,0.047,"Tue, 13 Dec 2011 00:17:18 GMT",78.46.47.67,Apache/2.2.9 (Debian) DAV/2 PHP/5.2.6-1+lenny13 with Suhosin-Patch mod_ssl/2.2.9 OpenSSL/0.9.8g mod_perl/2.0.4 Perl/v5.10.0
1.403,0.032,"Tue, 13 Dec 2011 00:17:19 GMT",78.47.156.38,Apache/2.2.16 (Debian)
1.403,0.313,"Tue, 13 Dec 2011 00:17:20 GMT",78.47.244.184,Apache
```

There is probably room for improvement here by automating the analysis of these results, however care must be taken into drawing conclusions from this data. As the example above demonstrates, there are multiple factors that must be taken into account. For example, NTP, which might have been used in the case of the last few entries.

3.4.7 Step 7 Gathering information about IP4 peers

Introduction Another option to reduce the anonymity set of a eepsite is to analyze your current neighbors on IP level and see if you can link certain properties of eepsite with regular IP addresses. Although this method is impossible to use when you have no indication of a possible link at all, it could be very useful to confirm a certain presumption. For example when you think a certain IP address hosts a certain eepsite, you might be able to notice that the eepsite is only online when the IP address responds to ping queries. Although this gives no hard proof of a link, it does increase the probability that a link between the two exists. Because we couldn't draw conclusions from the server string method in a way Crenshaw was able to, the gathering of this information couldn't confirm strong suspicion such as described above.

Looking up countries One of the properties associated with a host that might help in reducing the anonymity set is the country where the IP is registered. For example, a site in a certain uncommon language might be linkable to a IP address in a certain small (few nodes) country.

Crenshaw used the IPNetInfo tool to lookup the countries and uses SysExporter to convert the data tree to a plain text file. We've tried this, but though the IPNetInfo tool was very slow and couldn't handle such a large amount of IP address and the SysExporter tool didn't manage to extract the tree view from the IPNetInfo tool.

In search of a better alternative we found the GeoIPlookup tool in Linux which uses the Maxmind GeoLite database to lookup the countries. We wrote a script to automate this process, which gave us the country distribution in very little time. At first a lot of IP address were not resolvable to a country, but after updating the database version received by the standard Ubuntu repository (2011-01-15) to the latest version (dated 2011-12-06) from the website `http://www.maxmind.com/app/geolitecountry`, the hit rate was as much as 100 percent.

Country distribution of IP peers In the pie chart below you can see the country distribution of our IP neighbors.

3.4.8 Comparison Crenshaw's and our results

A summary of the results of Crenshaw's research and our research can be seen the table below.

Table 3: Comparison of results Crenshaw and this project

	Crenshaw	**Us**
Address book entries	1538	2113
Active eepSites	112	28
IP peers	1099	395
IP peers with webserver	172	373
Matches of IP/I2P addresses	21	0

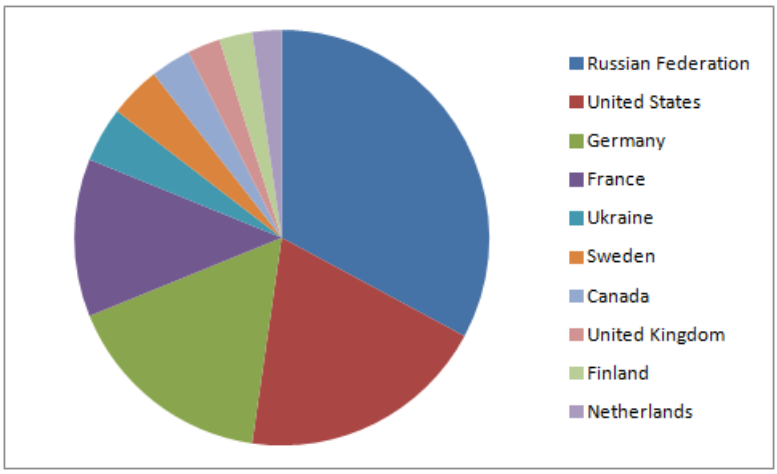

Figure 4: Country distribution of IP neighbors

3.4.9 Possible improvements

- Using another user-agent string might result in more findings.
- Use data gathered with zed attack proxy in an automated way.
- Store found information in a database, this helps deal with doubles and helps keep track of systems that are sometimes offline.
- We get a much better view of the I2P network if we store our findings in a database and run the scripts periodically.
- We can do statistics on the information and might learn the times most systems are online etc. etc. (Perhaps that fact alone can help us match IP's with I2P systems.)
- It might also be more helpful in comparing data between various scans etc. etc.

3.5 Evaluation of reproduction

The method was hard to reproduce, there are many intricacies of I2P that require tests to be ran many times over a longer period of time. There is room for improvement in the automation of these tests and the analysis of their results. Many eepSites have "blank" headers, it is unclear if this is standard in the eepsite server software or that users have manually configured their webservers for this. We have not been able to gather enough information to make a single link between an eepsite and a webserver listening on an IP address with the method described in Crenshaw's paper. The paper itself might be the cause of this, people probably have become aware of leaking certain information with which their I2P address could be linked to their IP address and might have taken steps to protect against this.

4 Improving

4.1 Introduction

To improve the completeness and accuracy of the identification of I2P sites, we proposed the following methods:

- Metacrawling: Crawl the I2P sites and collect files that may contain metadata. Scan these files in order to reduce the anonymity set of the author.
- Enlarge the set of IP neighbors.
- Enlarge the set of eepSites. This can be split up in several different items:
  - Try to locate other addressbooks that may give us new I2P hostname to I2P hash mappings.
  - Crawl known websites (such as forums) for links to (yet unknown) I2P hashes that may contain eepSites.
  - Recursive crawling the internet and I2P for new eepSite addresses.
  - Using search engines to find new eepSite addresses.

Due to time constraints we choose to limit our research to enlarging the set of IP neighbors, crawling for metadata and crawling for eepSite hashes in active eepSites gathered in the improvement part.
Note that only after some time (when reading about the protocol and architecture) we realized that only getting to know I2P hostnames was useless, because these are merely shortcuts (of local significance) to I2P node hashes. These methods are describe in the paragraphs below.

4.2 Metacrawling

4.2.1 Introduction

To improve on eepSite identification by means of configuration errors, or maybe better to say sloppiness of administrators, we thought it would be interesting to scan hosted files for metadata. This metadata could give us information, which could decrease the anonymity set of the creator of the file. Well-known examples of these meta data entries are GPS coordinates and camera brand/ model in pictures and owner names and MAC addresses in Office documents.

4.2.2 Finding the right tools

When developing a strategy for collecting these metadata, several applications were considered. We have looked at a well-known Windows tool called Foca. A free version of this application is available for download. This tool is a combined data crawler and metadata extractor. Although we think it a powerful tool, it has two disadvantages, which made us look for other solutions. The first reason is the restriction of the free version to only use two concurrent connections, while a big performance improvement can be gained when using multiple threads because of the high latency, slow connections I2P offers. The other one being that this application is a closed source Windows application, while in line with our education's motto, we felt more comfortable working with open source applications, preferable on a open source operation system.

4.2.3 Changing strategy

When searching the Internet for an alternative that meets (recursive crawling of I2P sites, so it should be able to handle a the I2P proxy, multithreaded and meta data extraction) our needs this turned out to be more difficult than originally thought. Moreover, we were unable to find a ready set application. Because of this we changed our focus from locating a meta crawler to finding just a crawler to first mirror a website locally and then scan the files for metadata. Even this (at first) turned out to be more difficult than one would expect. We have tried the following crawler, which were not suitable due to following reasons:

- ZAP (Zed Attack Proxy) - Used by Crenshaw to spider through I2P sites, this proxy can be chained to the I2P proxy to capture all requests. Once a site has been identified by ZAP, it can subsequently be spidered by this application. This means, you first have to visit a eepSite through the proxy and use the GUI of ZAP to further download the site. This seemed fairly cumbersome, and the GUI didnt provide the means of automation we wanted.
- Harvestman - We spend quite a lot of time trying to get this application to function, but somehow it kept downloading only a few files of every site for no reason.
- Yacy - At first we thought this might be useful, but it turned out to be a distributed site indexing crawler instead of a site mirroring crawler.

During the search suddenly a classmate noticed our effort and swiftly mentioned the reason why the crawler we were trying to find didn't exist. It was already built-in in every Linux system; using Wget recursively. To tune Wget to optimally perform for our situation, we used it with the following parameters:

```
wget --tries=5 --connect-timeout=5 --read-timeout=5 --directory-prefix=site-mirrors/
--random-wait --mirror --no-parent --no-verbose --no-check-certificate
```

4.2.4 Choosing our meta data extractor

To extract the metadata from the locally mirrored websites, we choose the following tools. All of these are open source, publicly available. These tools take a filename as input and give all metadata they are able to find as output.

- pngmeta (for extracting meta data from PNG files).
- pdfinfo (for extracting meta data from PDF files).
- extract (support for MP3, OGG, JPG, GIF, PNG, TIFF and PDF files).

The tools all give information for a specific file type, except for the extractor tool which gives information for a lot of file types, although being les specific. For these reason we created a bash script, which crawls recursively through the local mirror directories of the I2P sites, choose the best tool for the specific file, extracts and parses the information and then creates a transaction to store all meta data in a local SQLite database which then subsequently can be queried for useful information.

Note that we also scanned the results for the existence other file types that might have been usefull. If any of these would have been present, we would have extended our script with an appropriate tool (if available) Therefore also scanned for the following extenions: mkv, avi, mpg, doc*, xls*, ppt*, but there were no files of these types present.

4.2.5 The crawl

The set of sites identified while reproducing was used to crawl metadata. Of these 28 sites online at time of reproducing, 25 were online during metacrawling. We found out soon that running multiple Wget processes simultaneous made the I2P client crash quick and often. To avoid problems we adjusted the script to run only a single thread and restart the I2P when needed. The crawl took around 20 hours, starting at 12th of December at 22.00 till the 13th of December 19.30.

4.2.6 Results

The results of this crawl can be viewed in appendix 1 and 2. The first one give a summary of file types per site that could potentially contain meta information that is extractable by our tools. The latter one shows the information extract by these tools from these files. Because the amount of information is rather limited, it is quite easy to oversee the results. Most of the information found is only the tag of the image software used to create or edit the image. Also, you can see that the theme used on the Wordpress installation of http://hq.postman.i2p is created by someone who goes by the name of "Ducky" but since this is just the standard theme provided by the software, it says nothing about this specific instance. At http://ugha.i2p one image is tagged with what seems to be a real name and actual Internet address. But, since it also tagged with the year 1996 and the image itself is a small button with the text "New" this image is probably just downloaded from the web.

4.3 Enlarge the set of IP neighbors

4.3.1 Our method

The method we used, is a follows; we used Crenshaw's script to extract the current neighbors from the local database and scheduled this one every hour for exactly 3 days long. We then parsed the resulting data to remove duplicate entries per timestamp (which wasn't done by Crenshaw's implementation) and analyzed the data in terms of cumulatives and delta.

Results in terms numbers
In this picture you can clearly see that the number of concurrent known neighbors constantly keeps increasing, and there is constant number of peers who have been online for the complete three days. Stable nodes are more likely to host something in contrary to nodes that come online and leave short period later who are most probably just using I2P to send or receive something. The red line indicates a constant change of neighbors every hour. We couldn't find a reasonable explanation for the sudden drop at the end of the graph, but suspect it is rather simple. Because the number of neighbors seems credible (and in line with the other measurements) at the timestamps just before and just after the drop, we suspect the cause to be more than a simple hick up in the script or even maybe in the I2P client. The lather suspicion is based the on the fact that we noticed the I2P client to be rather instable, crashing every few days or so...

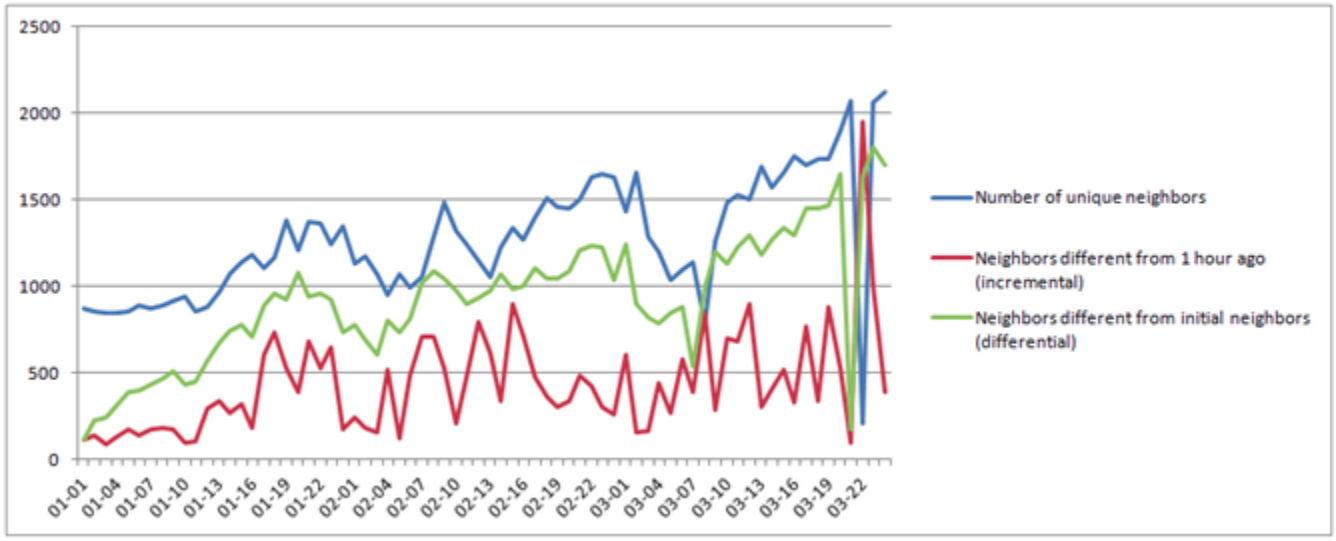

Figure 5: Data of 5,6,7 December

4.3.2 The results

A total of 101 different countries were identified using this method. For readability, the top 10 is displayed in the pie chart below. The full list is available in as an appendix.

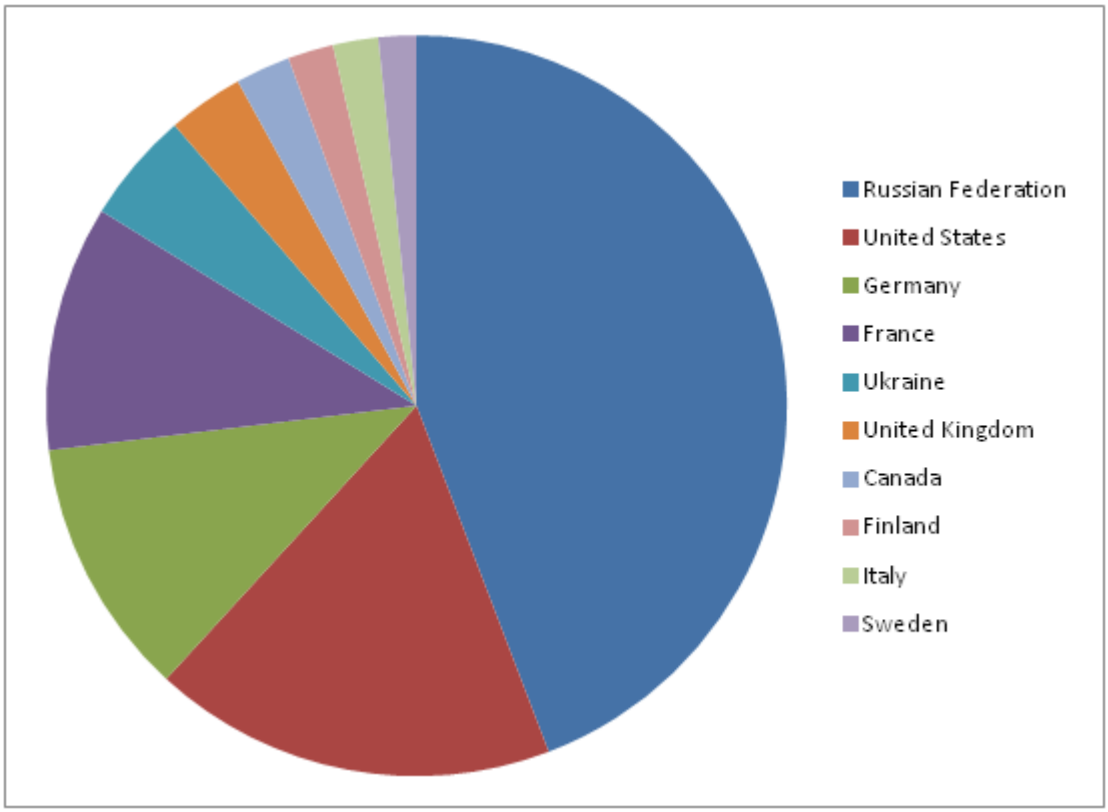

Figure 6: Country distribution of I2P neighbors

Besides reducing the anonymity set of eepSites, we thought some conclusions might be eminent from the data as such. Drawing these conclusions from this data seemed not that complicated at first, but after diving into it, it turned out to be almost impossible. Some of the ideas we had which might show up in the graph were:

- Did the number of Russian users increase during the election period?
- Do countries with repressive regimes show up with high numbers in the chart?
- Did the number of French users increase because of the 3 strikes law?

The reason we couldn't confirm nor deny these assumptions is that the data available to us was just to few. We only have data from one moment, one year ago. Moreover the country distribution of Crenshaw was based on a lot less peers (so less accurate) and the three strikes law turned out to be more than 2 years old. After some searching the Internet we also found messages of China actively searching for and blocking Tor peers[13]. Probably in a similar way like we are determining I2P peers. So even linking regimes to the number of users seemed impossible. All in all, analyzing this data as such turned out to be a completely different statistical research, which was out of scope for this project, so we moved on to the next step.

4.4 Crawling for I2P hashes and URLs

4.4.1 Introduction

Now that we have a mirror of these I2P sites on our server, we were able to scan these files for I2P hashes referenced at some forum or comment page. Before starting this project we also thought about crawling for I2P URLs, but this turned out to be not useful, since a I2P host can only be reached with knowledge of the hash of that specific host. I2P URLs are merely used to simplify browsing, using local host files to map I2P URLs to hashes.

4.4.2 The scanning

For the scanning, we used the following command:

```
find . \( ! -iname "*.jpg" -o -iname "*.png" -o -iname ".gif"\) -exec egrep -o ".{1,20}
.i2p[=][a-zA-Z0-9~-]{516}" {} \; > ../hashes.txt
```

4.4.3 Result

A list with the hostnames and beginning of the hashes (addressbook entries) can be found in appendix 3. A summary is displayed below:

Table 4: New found hosts name - hash mappings

12chan.i2p	planb.i2p	pb.i2p
4chan.i2p	job.i2p	trance.i2p
eepsites.i2p	onionb.i2p	forum.i2p
bl-tools.i2p	onionib.i2p	planb.i2p
irc.echelon.i2p	oregano.i2p	yape.i2p

4.4.4 Identifying these new hashes

To find out if the hashes found during crawling were new or were already in our address book, we first made a list of all I2P addresses represented by the hash. Next, we opened our host file and grepped for these I2P addresses. If we found one, we compared the exact hash of the host from the host file with that of the crawled hash to see if they were the same. We took this approach because I2P addresses do not have to be unique for a certain hash alice.i2p can coexist with alice.i2p because they can both point to a different hash, a different I2P node. The only constraint is that these two alice.i2p addresses are not in the same address book. If we did not find the I2P hostname in our host file, this entry would probably unique. But to be sure we had to test if the corresponding hash did not occur in our host file. The reason for this is that a certain I2P hash can change or have multiple .i2p addresses.

The results of checking for occurrences of I2P addresses in our host file were:
One odd occurrence, that demonstrates that our approach of also checking the hashes was necessary was the search.oregano.i2p and oregano.i2p hosts. search.oregano.i2p host does occur in our address book:

```
cat hostfiles.txt|grep 'oregano.i2p'
search.oregano.i2p=ocgVdC1LKML6atL5ucp6kn8AZJLD2RTTX8knRZ6zKufYfsHmeQSWtoYom6p4y6hDocqn1eJmBwBOo
AyhlRXUieHHkUEWDFySzyAFMRwIOHdEXkz6kVw6mvxBbLR35bNniTWTmIbaVMjZoT7VdVpJOsunVSzARnSSeiYqlSEuMxyUPh
X1wrfWIvzkBrW3-iwMUxTX677h9J9W5ZNwo9NPm6ZhQOPnY2PhJKx7ZCxQqcUA~bw4b53~ZW7tiYLSwdY82sclXiZcQkxJjma
IssuE5wZXVMOc9FiLz4MqfsJYlmvONM1wqsyVcPPnuRazsPhhuu52cBP8erY6U3KMtB5ShEZJLHoU3aU2DRaNPUr810XZ8W~i
NbbfAOonzuYnFY6yJ72gnrdlSt0r6nYlljb4Nd3yfPxS3AObZrA~gHVEI31hmUU888jV8R31JPv7g5nOpd4dSrMS6ne7bJlpE
2QotlMTKEaROxiR5Wdi6e2CJWNtzyp8SFPG1ON8DOhc23xAwBIU
```

Table 5: New found hashes

I2P address	address in host file	hash in host file
12chan.i2p	yes	yes
4chan.i2p	no	no
eepsites.i2p	yes	yes
bl-tools.i2p	no	no
irc.echelon.i2p	yes	yes
planb.i2p	yes	yes
job.i2p	yes	yes
onionb.i2p	no	no
onionib.i2p	no	no
oregano.i2p	no	no: search.oregano.i2p does exist, but with different hash
pb.i2p	no	no
trance.i2p	yes	yes
forum.i2p	yes	yes
yape.i2p	yes	yes

While we found search.oregano.i2p while crawling, oregano.i2p does not occur in our address book:

```
oregano.i2p=x5X2OFcVCxPZ1USdlUuLfzZerNSv2NWmlIRP2yrjAr-Lvis7FHrDAnMMZ-fVzsliqiP0dDY~VzbwNixt
6VEKJHN8STRlb9gwIK38zxfxO40fpDQMkHRwjUjtHuLmUbgMKHybuOzPDN1n3H18Lw0uJg83OwnR7ELX8HLPEowd~lfd
NwfJzJ9d0sKSa4MTF568Rd-ac3goijtE92Fxb2HhODyJ6RlSxgQUWWY-L6C6nJnEcRoAbrOzCo5-sDobc3iVu-BlXM5dm
xvb-udw95Z-ZGfc1erNC0VaNpt0EdQG85UOoDn8d4VQMzNBxF1O5F7tlTOqnq2JwSkwnjb0VE-KhuH2-YzTLBfZCFLrfp1
ZxZ1--Q9zIlJmzHCQAWUJspcnrX99RbjAGDpRXKSddYGA2zHVL11xt5nw-~Y~sTLvNjcigZiITk7hVWjhaWm33CIMvWnZf
rNO7wYx3Q1qtf2ofiYSKA2uALosJKjpQqOpdgxvuD6K8djeSazwgktNwRwAAAA
```

To verify if these sites were still active we had to add them to our hosts file. For some reason our new host file was not loaded so we added the hashes manually.
The following hash triggered an "invalid base 64 destination" error:

```
bl-tools.i2p=5CcqnvYZLkCgFHxm30oG-p0dwC9K1i97UOwZ93Hm61ak4zQnMChbZIEtlsjU60POOMAEFZccbdtm1fB99FU
b3A3doG~IQQqZ2yo8t3STNt0TQhePQKysQp9LlgeJlT5IvL9KAQGg7ZrHVPvkNU9GlRvHbN4Q7NIbEDrjHDkqcVR3dPTWHZn
xNRSVtb9ZUkmdfCpDCljt0yByubrWPBk~DX80YhAHNvDLOV15JXXrhw5oKVb0GZcBvCDaS9WDkyWPGexGXcaWO3Hzh8GQNDRN
u8FdwEcdVywmXhVqDXBuZrj1SU5jtNtIZpK8hsDSOyVoA6oSQ0WKCUazVPYi32By6XsXSiEANaKeQqKYQbGmLAmG8UmT6yp5K
1sy3kq0Y1JuTjcDdekAcglqU457VARtG7YbFbN7W480JTnIjyP8KJMCMzlYMDWZK5u1UPcIDXfQ~DEYd1KY~Qjb~SgPBZ8VzuK
yvOu8SKs9Ff4lFw3srBghzKGN3RvV
```

We started to manually verify the availability of the addresses, however after browsing to the first site '4chan.i2p' we quickly found out that this was not a good idea when a possibly illegal image was loaded in the web browser. Because we did not want to have anything to do with illegal activities on I2P, we decided not to continue with these new hashes.

4.4.5 Results

We have found 6 I2P addresses/hashes that are not published in the public address books we mentioned. These six I2P addresses/hashes can be found in the appendix. (Note that one hash was invalid) These I2P addresses/hashes could be added to the host file after which the method of investigation could be repeated to see if more I2P addresses/hashes, metadata and perhaps even correlations between IP addresses and I2P addresses could be found. For the reason mentioned, we decided not to.

Table 6: Unique new hashes

I2P address	**status**
4chan.i2p	online
bl-tools.i2p	?
onionb.i2p	?
onionib.i2p	?
oregano.i2p	?
pb.i2p	?

5 Conclusion

Unfortunately we weren't able to set a baseline the way Crenshaw described in his report. His method was mainly based on comparing the banners of webservers that are accessible over both the I2P network, as regular Internet. He then used other detected properties to further strengthen his suspicion of a certain link. Because we could not link any webservers based on their headers (most were anonymized), enlarging the set of IP neighbors wasn't very useful.
Crawling eepSites in order to find more hosts has proven to be a worthwhile method; we found a number of hosts that are not in public address books. Moreover, when trying to enlarge the set of eepSites, we stumbled upon possibly illegal material, on which we chose to stop further digging into this darknet. Also with the scanning of metadata, we weren't able to pinpoint the identity of specific sites. The amount of information found indicates that most webmasters are probably aware of the fact that identifying information might leak through metadata. One could however argue that through identifying used image software, we were able to reduce the anonymity set of some sites.
Furthermore, we think that choosing a specific approach instead of a generic approach might give some extra information. With this statement we mean that if you target one specific site, you can dive deeper into it. For example, specific tags in the HTML files or maybe an author statement in a CSS file might give clues on the identity.

6 Evaluation

Before we started this project we thought that setting the baseline through reproducing Crenshaw's research would only be a small part of our project. But, during the course of the project, we stumbled upon lots of problems; there were gaps in the paper we were trying to reproduce, some scripts didn't work correctly, for other scripts the input was not clear and so on. Because of this, the establishing of the baseline took a lot of time, which we had rather spend on finding improvements. This also had to do with the fact that the subject is completely new for us. Because of this, we spend a lot of time on reading about the architecture. Furthermore, because the banner grabbing didn't give us any conclusive information, other steps we took were rendered rather useless. We found out later that the I2P community was made aware of these information leaks by Crenshaw. A new I2P version has since been released that strips header information from the webserver that ships with I2P.
Another setback was the possibly illegal material appearing on our screens. We didn't count on this because we thought we wouldn't stumple upon it that easy. We expected this material to be hidden deep inside the network. Somewhere were you would have to put a lot of effort in establishing relationships with other participants before you would get access to the hashes of these sites. We decided to absolutely go no further on this point. This decision could be argued; we stopped efforts of de-anonymizing when we found illegal material, but we potentially could have gained hints on who was responsible for this material. We made this decision for the reason we don't want to be associated with this material in any way. We dit notify the relevant authorities of our findings. To conclude we can say despite these setbacks we learned a lot while doing this project. What we especially liked, was the fact that we could relate a lot of the concepts to the SSN and DIA courses of the SNE master to this subject of research.

7 References

References

[1] Adrian Crenshaw: "Darknets and hidden servers: Identifying the true IP/network identity of I2P service hosts"

[2] Juan Pablo Timpanaro, Isabelle Chrisment, Olivier Festor: "Monitoring the I2P network"

[3] Michael Herrmann: "Privacy-Implications of Performance-Based Peer Selection by Onion-Routers: A Real-World Case Study using I2P"

[4] Peng Deng: "Anonymizing Networks"

[5] I2P websites: `http://www.i2p2.de/techintro.html`

[6] I2P websites: `http://www.i2p2.de/i2ptunnel`

[7] I2P websites: `http://www.i2p2.de/how_tunnelrouting`

[8] I2P websites: `http://www.i2p2.de/how_intro.html`

[9] I2P websites: `ttp://www.i2p2.de/netdb_discussion`.tml

[10] I2P websites: `http://www.i2p2.de/naming.html`

[11] I2P websites: `http://sempersecurus.blogspot.com/2011/06/i2pthe-other-anonymous-network_18.html`

[12] RFC 2396: `http://www.ietf.org/rfc/rfc2396.txt`

[13] TOR in China `https://blog.torproject.org/blog/tor-partially-blocked-china`

8 Appendix 1 Files found during site crawl

Table 7: Found file types per site and extension

	jpg	**png**	**gif**	**tiff**	**pdf**	**mp3**	**ogg**
bl.i2p	1	0	0	0	0	0	0
complication	1	1	0	0	0	0	0
diftracker.i2p	5	4	7	0	0	0	0
docs.i2p	0	0	1	0	0	0	0
echelon.i2p	0	0	0	0	0	0	0
eepsites.i2p	1	0	11	0	0	0	0
forum.i2p	0	0	0	0	0	0	0
freshcoffee.i2p	0	0	1	0	0	0	0
hq.postman.i2p	4	0	0	0	0	0	0
i2pbote.i2p	15	10	3	0	0	0	0
i2p-bt.postman.i2p	0	4	0	0	0	0	0
i2plugin.i2p	0	0	0	0	0	0	0
i2p-projekt.i2p	0	20	0	0	0	0	0
mpaa.i2p	2	0	2	0	0	0	0
perv.i2p	0	0	0	0	0	0	0
planet.i2p	0	0	0	0	0	0	0
redzara.i2p	0	0	0	0	0	0	0
riaa.i2p	1	0	0	0	0	0	0
stats.i2p	1	2	0	0	0	0	0
trac.i2p2.i2p	0	19	2	0	0	0	0
ugha.i2p	0	15	0	0	0	0	0
update.killyourtv.i2p	0	0	0	0	0	0	0
update.postman.i2p	0	0	0	0	0	0	0
www.i2p.i2p	0	27	0	0	0	0	0
zzz.i2p	0	0	0	0	0	0	0
totaal	31	102	27	0	0	0	0

9 Appendix 2 Found meta information

Table 8: Metainformation: http://hq.postman.i2p

/wp-content/themes/default/images/kubrickheader.jpg	comment	DuckyF
/wp-content/themes/default/images/kubrickfooter.jpg	comment	Ducky <
/wp-content/themes/default/images/kubrickbg.jpg	comment	Ducky <
/wp-content/themes/default/images/kubrickbgcolor.jpg	comment	Ducky <

Table 9: Metainformation: http://i2pbote.i2p

/images/de.png	software	Adobe ImageReady

Table 10: Metainformation: http://i2p-projekt.i2p

_static/images/fr.png	software	Adobe ImageReady
_static/images/nl.png	software	Adobe ImageReady
_static/images/us.png	software	Adobe ImageReady
_static/images/it.png	software	Adobe ImageReady
_static/images/cz.png	software	Adobe ImageReady
_static/images/es.png	software	Adobe ImageReady
_static/images/de.png	software	Adobe ImageReady
_static/images/ru.png	software	Adobe ImageReady
_static/images/zh.png	software	Adobe ImageReady
/_static/images/tabletitlelight-tall.png	software	Paint.NET v3.5.1
/chrome/common/draft.png	software	Adobe ImageReady
/chrome/common/desc.png	software	Macromedia Fireworks 3.0
/chrome/common/ics.png	software	Macromedia Fireworks 3.0
/chrome/common/edit_toolbar.png	software	Polybytes PolyImagePro
/chrome/common/feed.png	software	Macromedia Fireworks 4.0
/chrome/common/asc.png	software	Macromedia Fireworks 3.0
/chrome/site/content_background.png	comment	Created with The GIMP
/chrome/site/file.png	software	Adobe ImageReady
/chrome/site/folder.png	software	Adobe ImageReady
/chrome/site/tracnavbg.png	comment	Created with The GIMP
/chrome/site/navigation.png	comment	Created with The GIMP
/chrome/site/find.png	software	Adobe ImageReady
/chrome/site/feed.png	software	Adobe ImageReady
/chrome/site/parent.png	software	Adobe ImageReady

Table 11: Metainformation: http://ugha.i2p

/moin/modern/img/draft.png	software	Adobe ImageReady
/moin/modern/img/moin-new.png	unknown	(c) 1996 Leo Doerr @ www.silverpoint.com
/attach/IgnaciosToopie/attachments/i-toopie_logo.png	software	Adobe ImageReady

Table 12: Metainformation: http://www.i2p.i2p

/_static/images/fr.png	software	Adobe ImageReady
/_static/images/nl.png	software	Adobe ImageReady
/_static/images/us.png	software	Adobe ImageReady
/_static/images/it.png	software	Adobe ImageReady
/_static/images/cz.png	software	Adobe ImageReady
/_static/images/endToEndEncryption_zh.png	software	Paint.NET v3.36
/_static/images/es.png	software	Adobe ImageReady
/_static/images/de.png	software	Adobe ImageReady
/_static/images/ru.png	software	Adobe ImageReady
/_static/images/zh.png	software	Adobe ImageReady
/_static/images/tabletitlelight-tall.png	software	Paint.NET v3.5.1

10 Appendix 3 Hashes found during crawl

12chan.i2p=u5p9cLfKFqjUz5xtZJN2YWC6C2IKu1cKBFVj7WcZLnsQM68oxWJpOBOAiWOeV2h2sJf8HONf12PJxUO4NxkjuOxIiyEOySt JRMU6~tSyIJ1ozWumu AmxLtlZALgx~My0tJU73kU~AnR55gnJBXkVlywUVcKV5Y~186bXrARdqqNBLVHfZ-mUETnQkDVAnjoB-UZeKAkqYk2SLBpAAdTZGOmRmC89pVu1UIiBFrs8iVs3E TVBAUCr6X8d5Ly5yDSsSNB~-RWP3BllKVZc66xzWlTdp96ZMYsLqTAMn-N4~3GZ9sQQfBXVOjHbTm1rQjBe~6amAJJgx6s-4zS-Nmv~6AtrpEQhAfn-JCyxWkJoG zpAqqJjEg-CdsaFw9RAarpln8HaN3mNexkaIzIx9-gFyA1pchYjjcbuoEPgzdfyPzEnfwUB~QM3cZcwGxQS1iaHgti6OV96a868sVajyEILNgTEJ4Hg8UQyoue02 izI2wW8FETX3vGjT2r1XLURFA5OAAAA

4chan.i2p=a8lxRe6FoDNuELp9q2TcUWgNhiRIn8zJCQArHblWvWOA2nZKnDltjGS2F3s0yGl~XOtYzG8DdjUxVX7A8oV9DsyLasiDGrzFdgIqoY6MxreWzzdc4o 9NfznMgWOU10aaUmzFC7vYm4tYL-g02jbzb6MQYUfxnpJYN6qJPbq-dVEleSEwbp69Q3lt7NjlCRKZ5UnmQbWJ~saVqBslNJ~KRup2PE89V0EUdZvrt3P~3HObAL bZWETGso4rJtaBFZMfO57ptTGGf47~tH1ZVhH2eGqcGbw9XcHriItveywlyxpOQLOBD8orMcfgWed1i8iD7M9ovTarfOnY2mgeEuX3OElcMu0IxzMhQZzWoJdeeM Gg1U4qkQhwOruZi0IePnYf6SsGa25W7pPtxEPLsNn912wtrOPqPm~l1I6BiGfIihyF-oSoOqTwgYmg4hs5U-jvz~zMy6tfGBz8K5NX4p~ys99fVedpKh4r9-32Gz vpcIpeP5XwVE3BdOjHHElkOQ-IAAAA

eepsites.i2p=DHoGpNRACpcGnhB1l14ZRQFjnQtNvItX7~bZylgVaD2DnaYpqB7GoBcf5gJmjhXYRi3-UlOnrkEtQB4fICotgMOXLG8LHyNQVkj6rAnGSWBwX3z ZaTWr~K6Up80AMDa8nN3DO4coXQiL8Jn4~pcN2-7pxIL-Ox-J3aB8f3KthVxcuR27h1fJGR~BFqf~4ZdC7rGItokOb0qocU4gMNI-HWgmhUv-ucFc9m~Jh-BvM2n d~KGs7EQMtRxLgVJcj7GOj9chyN37l7mB5X~vYIueORENcKgo3OTjTAE9oHiVSZz7v11VVH3OynUp65QdtVXLCrqsqr4nJoFnApUcX5uYY4h~24dthZfHv1TjzWO uy1DFJH5mmT81HGQhtQI8g52CbnRgm4ItKQNq8AYPkJI2habi4ibUqBnmYk6gOp9mx1MsmpMmbQP6jemiZPpcVhAEjqJO-IyPyyOpDvg6O4fIgmstwdQS~qq~Sts 2OBWikhsXifQL87wnXtiN9e-PluyNAAAA

bl-tools.i2p=5CcqnvYZLkCgFHxm3OoG-p0dwC9K1i97UOwZ93Hm61ak4zQnMChbZIEtlsjU6OPOOMAEFZccbdtm1fB99FUb3A3doG~IQQqZ2yo8t3STNt0TQhe PQKysQp9LlgeJlT5IvL9KAQGg7ZrHVPvkNU9G1RvHbN4Q7NIbEDrjHDkqcVR3dPTWHZnxNRSVtb9ZUkmdfCpDCljtOyByubrWPBk~DX8OYhAHNvDLOV15JXXrhw5 oKVbOGZcBvCDaS9WDkyWPGexGXcaWO3Hzh8GQNDRNu8FdwEcdVywmXhVqDXBuZrj1SU5jtNtIZpK8hsDSOyVoA6oSQ0WKCUazVPYi32By6XsXSiEANaKeQqKYQbG mLAmG8UmT6yp5K1sy3kq0Y1JuTjcDdekAcglqU457VARtG7YbFbN7W480JTnIjyP8KJMCMzlYMDWZK5u1UPcIDXfQ~DEYd1KY~Qjb~SgPBZ8VzuKyvOu8SKs9Ff4 lFw3srBghzKGN3RvV

irc.echelon.i2p=ooAkOd014ooZXQ4rpvFsqAZ108YU9knllXDocY5fD84IrRVXDbRyUFtlboDmiKDPNRTY2JIXvb-l5FahAB9SUAZ8voAdH4ozHrigHVg6scVt wU7GCfjq5byuwnyemupgk6saBQKfuyc7k3bYl5Iro85Lnx6EnaYYbyWjMwoUWksI6o3Hp5bNyj2B5wHX7HLzeg1ByxQZ4Q8BZbALg8yIcvKn3rHN4tO9FhQ-e5u6 hRxINbMtBL1Hasxl9I8XSI0yalKSsmpWDNxRcx-2VKlVh3MPK~9VZlBrQzXwObLmpTuTukeNt8nheFQ7mZVAHt~1gCs2TmjQPVm2g9BQ32zqCFDhePNfBcxcG-Xl 56mE9n6kUvaJWsy5~VhiZZmCUGKWMx0uH7odSiC3ohNgv7dhDpANVA~gvt0Ihhwlnif HBa~HHfjFQz75kl0nzv3htkcBfLt4M3AZ5bKLP8ymeGP5NYINf6uKNJA9 XrSoBjmv0GqvBvDP0Zl5OO2tkIxOBEWfAAAA

planb.i2p=SnDT-8j94Ot9DO--3KkfkFVhIyMypo~d3rPrXjOzScRW2ZSAle1T1MEr9E5MEYXNnT3PW5d9rU4R6QnhjM2MB~OuXOq83R6yngiL-4xncG2rcZQjfh ASvb91eSARuUmZQXsXg7gjuuKqLSIg8X-NFUnWapvnnx8E-cB4u4QE1zGnbo33Kt2wX3k31XyxzZwJmK-b0kYE41KioET~dOmFmP6oEvbAQPlq5h3YDNniyrAncy 4321pYwCde1KS8uIp3d9OEOc3u2KJAbsAf~KFhWj4~MEI5tPep5ti7iWm7lcnpsXxeeFDl4u5peEgUmu4bpGzyooux4~jpvQzgYv9NSmuFK9vfCA6nXs81bmHUJ4 qLUz~pMCACapXWxUOzDhE8hCABPbCJG8EX8k2JkO-fA4L1NhnF6MIrs4CPqlmyiLnBRXOdWq1AEMe1huTuodwDOtczFkJeDpRoHBoLQKPHbSZWShDzzdjuQ97VUC 5Vwm8C~ebFIf~UXmpBbzpTGQVqAAAA

job.i2p=sDK28lemTuN93PGWPeR7IBOFeLyRe-AU9xn7bSebufevzVvo8wOUfQFpBvlgtWQ5nvKNLSHxBZiOwK4wOsYHf6inlmU48DjD8UgHeSwvmiVROHctQ2jP EFP-ZCMsyDVTDLpsvhox4rAQTEHpfY1kYQSSc-OPnoXNsA~VIyOSdYMLXLwjTgOgCoJO--VTAHyIOSMJCTt7v5oBPRbICYB7KEwTD~6jqsCmilHpkMLHVt5uKpBj aVBOQrVEso4Z9lMMK~cK8IwAAtrfCT2PVFboXTIbIVAaSpNChyzG7iNrBHz7tDPLnGjrtpiOz7I7YK8wU64Uqb3epIvtR9ohyuTOYyS7NpDaf0ZL6n2cEfNjwnR-IUeRGWFtmRvc~lZREhir64ojquF82YRRkhwKGRQiqZdxcI-pJow-wtnnO5b1p-vToweRkociGMCCZISBHNHuivrOMz8z7Uw8BCiUdhLbpAFQxKdE31bq4cveblg8 9UAqmKnj9i3nSTiSmFjKoo0uAAAA

onionb.i2p=H~2NFWWBVHIwhBgIFwC-1XSAoR5tXjUw3Aa~PQK4LU2j7DCiLzbB3wI2nVG14XPQZ9MkShEhGwLbztKxtte7jY2eeC2mhYmQYK8XknYJRVTJlVS2E WSuKwbzc6Hx1YZLGBzf607jsCfei~EQCuSzwBVWkttCVRDkh7tCIfdZRlzOsKYkm9dr3NViBqmzARkleNYNjkT-RBXN6ed~JRKsuugfhD4-46n6UJIGmv5NT0NBo BPbteJjoMHtStB4ktow2KvLusMWx5gb7gAWIQgLuUc1QJKEaOKgjF8o8wI89peDnGIdO7aGC6qYDseZ5l34WTjvcArhMdEDytHUR0vauBVhgcGyCe9u~xOF3as0E vUM~aLXiNR5bRuj0ryrFCpZ4kZck6~--UaYiIcrRxqM~5pjqlNxDiIV4PTQ-~Xs5vBWvZypRnJx9poQfH-m5C2LZpyqqISTf~tkCKKQupK7b2OlqKTKAdYJI6xMn OsDOi1-FT7sn5EyCB1G8qaDhI6bAAAA

onionib.i2p=3tBtIi2xHT9hfehEx1Ea2UTQDxkaXZ6nsVSSfRsf5-VB3TxLQBQYJxFxOcyeni75wC2RQILi7kgPA00~jLiEIeTpP6nLpvxC8d~QrvyzA5F0UC7B 4ZzdgIgPU1kEAsAPTrE83x7-502MvQDhkAftrivuydWq~mCIsFvvTW57zS8JrUYlV2oCZo5-9QxtOgRbOPsCGv7gmKtSraJJP1WzUAg0CJbq91zDOrXpxVkn2Yc3 qpEfcH-DDxJtnT2pA9m6RYGhxD2lSB-Zm7ipwOTyDr1URB4xpwvw4jOWqxa6Ip9tA8plZ-V7WShD8nEAX3bFnSgDWBLXYQiZqyOutlOew44dAoYWE84JYduXlOqQ jWPc35ZuYcn7OSIMP-4frklgSlJgTAcAn---sryytD-1igLc456NmXa-6RzasbOCwBW4wEyCcWkaXXe8-B2VMGYXDNgf2H4o8exxeiSARatQn8fqkJrdGZOrUV7R ecGbh3miV72s0dgUhJPc5OJNQcziAAAA

oregano.i2p=x5X2OFcVCxPZ1USdlUuLfzZerNSv2NWmlIRP2yrjAr-Lvis7FHrDAnMMZ-fVzsliqiPOdDY~VzbwNixt-6VEKJHN8STRlb9gwIK38zxfx040fpDQ MkHRwjUjtHuLmUbgMKHybu0zPDN1n3H18Lw0uJg83OwnR7ELX8HLPEowd~lfdNwfJzJ9dOsKSa4MTF568Rd-ac3goijtE92Fxb2HhODyJ6RlSxgQUWWY-L6C6nJn EcRoAbrOzCo5-sDobc3iVu-BlXM5dmxvb-udw95Z-ZGfc1erNC0VaNpt0EdQG85UOoDn8d4VQMzNBxF1O5F7tlTOqnq2JwSkwnjbOVE-KhuH2-YzTLBfZCFLrfp1 ZxZ1--Q9zIlJmzHCQAWUJspcnrX99RbjAGDpRXKSddYGA2zHVL1lxt5nw-~Y~sTLvNjcigZiITk7hVWjhaWm33CIMvWnZfrNO7wYx3Q1qtf2ofiYSKA2uALosJKj pQqOpdgxvuD6K8djeSazwgktNwRwAAAA

pb.i2p=cLEewNwaPgr0NAB3JidRkWlz9NRlHasGygWyzWt6z2k0~H5JVnBzBNr26ZljgZgAsIpqOWrf~s7AFVceFppG1USUjzgLKvFomxWExUPxx4yrGpOBYzJPi BxKQWAdgofJ-4T-2s2xiOtx3nbbMVcTlJeGY9Px-k2yQJHvRmIMTCbF156Z7BZeAQ1fR-fqgbq-pHA~8rYpCrh69AuPRP06AcQTH3XFWf7JchQYn5qIr6zFjkWpe ZN-B5x9N1UD5YzoRifR1U4XlP8UcA1BUmZ3pg4DlryZKnWw6kvHUK63dSvvNWyZB~PUEWcYhrt6Lky2mOWL7oix3Vod7yb4CLOUOjc5gnCVPUk-hu7xdR9buTUnR TxxaoGjQVh1~NqdPOI~R1X4tvU1I3R~DjVdotXLmgqBnuZVwN-O-VrE1D4WTva4ggVoq2~UgSfZ7LrgeJahYkxaZSmRPLAkLSPqc0Y9njalS~90ErvRrC2uMpTzH T-wjY8~InGhWuIRL3gQWiGWAAAA

trance.i2p=5NDQ~tBi4KhlAE16dhoFoGO4daWxWPX69uUcuh-O74wFbzpKk9Z7Rsa-uB-~2qRYvjAQNqDIkt7Iigv-VYJuTRzX8AQsXuCRDzLDBLeGDoH5h6066 7DsfdnxuMcqXDYOt19XzoJ~Zh90yfPIn0zHNPz1YHqQAOHniPyGwz2nWSmh2OyS0c8-b1thZjIPRPoAL7Re4T-m87btKxz9FlYMcXiGcjWfzjSVwJH3RkJWRlotO ncF3SvOzpOEMVZv654VTiSxWYoeIkTffpsoOL4QWcpXFYQ-UB12nXlLJUMGPVbhoOOmJ4RDoyis9lcIhEX45N9WuSGpm-2z4O2LUY~Vf5UUihyWmjYAuEfaskhzO ak98d2oQOPZdDGyJkRF2pvZsYEFjL4BLaAaOOzpidieQ4Fq7O4nGCuCYbzbuWVMaU7uSKwk4aaIjIOZs6YvOLIiNy3xPts7OY9wkVKy4VZL8kTm~E4gUQ1FIaakM -tvk7rFnO5G0~i8bydd76iEPJM3AAAA

forum.i2p=XaZscxXGaXxuIkZDX87dfN0dcEG1xwSXktDbMX9YBOQ1LWbf0j6Kzde37j8dlPUhUK9kqVRZWpDtP7a2QBRl3aT~t~bYRj5bgTOIf9hTW46iViKdKO bR-wPPjej~Px8OSYrXbFv2KUekS4baXcqHS7aJMy4rcbC1hsJm3qcXtut~7VFwEhg9w-HrHhsT5aYtcr4u79HvNvUva38NQ4NJn7vI9OPhPVgb5gxkefgM1tFOQC 6QO1b~RADN~BW~X2S-YRPyKKxv6xx9mfqEbl5lVA1nBTaoFsN5ZfLZoJOFIVNpNoXxCrCQhvG2zjS-pJD2NF6g0bCcT4cKBWPYtJenLiK3L6fKJuVJ-og5ootLdJ NBXGsO~FSwdabvDUaPDbTKmqS-ibFjmq1C7vEde3TGo3cRZgqG0YZi3S3BpBTYN9kGhYHrThGH69ECViUJnUWlUsWux5FI4pZL5Du7TwDYTOBwnX2kTdZQ8WGSFl flXgVQIh1nOXpElShWrOQPROjGAAAA

planb.i2p=SnDT-8j94Ot9DO--3KkfkFVhIyMypo~d3rPrXjOzScRW2ZSAle1T1MEr9E5MEYXNnT3PW5d9rU4R6QnhjM2MB~OuXOq83R6yngiL-4xncG2rcZQjfh
ASvb91eSARuUmZQXsXg7gjuuKqLSIg8X-NFUnWapvnnx8E-cB4u4QE1zGnbo33Kt2wX3k31XyxzZwJmK-bOkYE41KioET~dOmFmP6oEvbAQPlq5h3YDNniyrAncy
4321pYwCde1KS8uIp3d9OEOc3u2KJAbsAf~KFhWj4~MEI5tPep5ti7iWm7lcnpsXxeeFDl4u5peEgUmu4bpGzyooux4~jpvQzgYv9NSmuFK9vfCA6nXs81bmHUJ4
qLUz~pMCACapXWxUOzDhE8hCABPbCJG8EX8k2JkO-fA4L1NhnF6MIrs4CPqlmyiLnBRXOdWq1AEMe1huTuodwDOtczFkJeDpRoHBoLQKPHbSZWShDzzdjuQ97VUC
5Vwm8C~ebFIf~UXmpBbzpTGQVqAAAA

yape.i2p=QyrT6wt7yvVuJhEH8k7G6u92~f4Ue5e7kDH2YB~J3GqQB6QiD5-11ffqQonAOpUxF7sLExS1tnWVGugY73IRr5Kfd7EI7EO1ELbnXfn-TuE51k5WuZD
-bZfVibjvjfx2oJ9d2n6Sh-mr59lF8bK1NlqmiPrOm~Ef0rfL3glpKGxEbptcOKrS8uh6UoI8kcU-zN1AVbSFuoPFj4tss9CDc6hFojnXEKmiGga3UJQJ8ss1SNj
VlaCz70XRfOfLT6YeZ7K57PuiCvhmY8-dudtgDdedJ~Al7qScQMe606JCpD6Gt0LVCNwTofIifhcBsjw7JNxzE4dTuqEPqtluIsumQ4aztPOmFw8kIM3cAbwhU4H
4PBPQBdlMBy9S3bSvbQd0lhDcLvs0dHyGyYJ9THx-ORj8f98AVXtcK0rJu26tiTJTyPjwXHvcAFSLZmITgbd7ppKhW44hN0iVwsebps03-nxgYyQ9XfLvNy2KIyC
QI2Jaz~6k0eStAUBRQa3pS2EIAAAA

11 Appendix 4 eepsite addresses with HTTP header info

"freshcoffee.i2p","200","Apache"
"ugha.i2p","200","blank"
"forum.i2p","200","blank"
"mpaa.i2p","200","blank"
"stats.i2p","200","blank"
"eepsites.i2p","200","blank"
"hq.postman.i2p","200","blank"
"www.i2p2.i2p","200","blank"
"i2p-bt.postman.i2p","200","blank"
"i2p-projekt.i2p","200","blank"
"perv.i2p","200","blank"
"i2pbote.i2p","200","blank"
"docs.i2p2.i2p","200","blank"
"i2plugins.i2p","200","blank"
"trac.i2p2.i2p","200","blank"
"update.killyourtv.i2p","200","blank"
"update.postman.i2p","200","blank"
"tracker2.postman.i2p","200","blank"
"echelon.i2p","200","blank"
"ugha.i2p","200","blank"
"forum.i2p","200","blank"
"freshcoffee.i2p","200","Apache"
"mpaa.i2p","200","blank"
"riaa.i2p","200","blank"
"redzara.i2p","200","blank"
"i2p-bt.postman.i2p","200","blank"
"hq.postman.i2p","200","blank"
"eepsites.i2p","200","blank"
"zzz.i2p","200","blank"
"i2p-projekt.i2p","200","blank"
"www.i2p2.i2p","200","blank"
"update.postman.i2p","200","blank"
"tracker2.postman.i2p","200","blank"
"docs.i2p2.i2p","200","blank"
"i2pbote.i2p","200","blank"
"pastethis.i2p","200","blank"
"trac.i2p2.i2p","200","blank"
"ugha.i2p","200","blank"
"forum.i2p","200","blank"
"freshcoffee.i2p","200","Apache"
"bl.i2p","200","blank"
"mpaa.i2p","200","blank"
"riaa.i2p","200","blank"
"complication.i2p","200","blank"
"redzara.i2p","200","blank"
"i2p-bt.postman.i2p","200","blank"
"stats.i2p","200","blank"
"hq.postman.i2p","200","blank"
"eepsites.i2p","200","blank"
"zzz.i2p","200","blank"
"www.i2p2.i2p","200","blank"
"perv.i2p","200","blank"
"i2p-projekt.i2p","200","blank"
"echelon.i2p","200","blank"
"tracker2.postman.i2p","200","blank"
"docs.i2p2.i2p","200","blank"
"i2pbote.i2p","200","blank"
"i2plugins.i2p","200","blank"
"planet.i2p","200","blank"
"diftracker.i2p","200","blank"
"inr.i2p","200","blank"
"update.killyourtv.i2p","200","blank"

12 Appendix 5 IP addresses with HTTP header info

```
"146.52.96.160","401","httpd"
"193.41.7.64","200","Apache/2.2.16 (Debian)"
"188.134.2.118","404","blank"
"188.134.81.69","200","nginx/0.8.54"
"194.14.172.60","200","lighttpd/1.4.26"
"212.114.254.91","400","nginx"
"212.12.40.162","200","Zope/(Zope 2.9.7-final, python 2.4.6, linux2) ZServer/1.1"
"188.40.181.33","200","lighttpd/1.4.28"
"213.232.95.57","200","Apache/2.0.52 (CentOS)"
"178.25.89.6","200","OmniSecure/3.0a5"
"178.124.156.184","401","RomPager/4.07 UPnP/1.0"
"194.88.206.104","200","nginx/0.7.62"
"212.232.42.95","200","lighttpd/1.4.29"
"213.156.229.195","404","Apache/2.2.9 (Debian) PHP/5.2.6-1+lenny13 with Suhosin-Patch mod_ssl/2.2.9 OpenSSL/0.9.8g"
"209.6.82.6","200","Apache"
"184.107.206.46","200","nginx/1.0.9"
"178.169.68.27","404","blank"
"184.56.202.50","200","httpd"
"173.255.217.222","200","Apache/2.2.14 (Ubuntu)"
"192.138.213.236","200","Apache/2.2.9 (Fedora)"
"206.253.164.169","200","nginx/1.0.5"
"106.187.39.171","200","Apache/2.2.14 (Ubuntu)"
"198.133.224.147","200","Apache/2.2.9 (Fedora)"
"213.155.17.24","200","Apache/2.2.16"
"217.147.161.13","200","nginx/0.7.65"
"46.4.248.202","200","nginx/0.7.67"
"217.210.127.18","200","Apache"
"213.170.70.61","200","Apache/2.2.17 (Win32) PHP/5.3.5"
"31.207.192.133","200","lighttpd/1.4.26"
"62.141.60.50","200","Apache/2.2.9"
"62.168.243.54","200","Apache/1.3.34 (Debian)"
"68.48.83.242","200","Apache/2.2.14 (Ubuntu)"
"66.108.39.87","401","httpd"
"69.164.196.4","200","Apache/2.2.17 (Ubuntu)"
"46.249.16.11","200","nginx/0.8.53"
"2.132.135.23","401","RomPager/4.07 UPnP/1.0"
"71.239.91.32","200","nginx/1.0.5"
"75.145.125.59","403","Apache/2.2.14 (Ubuntu)"
"75.145.125.61","200","Apache/2.2.21 (Unix) mod_ssl/2.2.21 OpenSSL/1.0.0d PHP/5.3.6"
"78.46.47.67","403","Apache/2.2.9 (Debian) DAV/2 PHP/5.2.6-1+lenny13 with Suhosin-Patch
   mod_ssl/2.2.9 OpenSSL/0.9.8g mod_perl/2.0.4 Perl/v5.10.0"
"78.47.156.38","200","Apache/2.2.16 (Debian)"
"78.86.34.19","200","httpd"
"77.64.153.140","200","Apache/2.2.17 (Win32) mod_ssl/2.2.17 OpenSSL/0.9.8o PHP/5.3.4 mod_perl/2.0.4 Perl/v5.10.1"
"77.37.172.127","404","blank"
"78.231.24.103","200","Apache/2.2.14 (Ubuntu)"
"78.47.244.184","200","Apache"
"76.250.48.108","200","lighttpd/1.4.28"
"82.103.129.142","200","Apache"
"83.169.5.75","200","Apache/2.2.8 (Ubuntu) mod_jk/1.2.25 mod_python/3.3.1 Python/2.5.2 PHP/5.2.4-2ubuntu5.17 with Suhosin-Patch
   mod_ssl/2.2.8 OpenSSL/0.9.8g mod_perl/2.0.3 Perl/v5.8.8"
"85.223.118.207","500","Apache/2.2.21 (FreeBSD) Phusion_Passenger/3.0.9 PHP/5.3.8 with Suhosin-Patch"
"81.25.48.117","200","Apache/2.2.21 (Unix) mod_ssl/2.2.21 OpenSSL/1.0.0e DAV/2"
"85.228.251.54","200","Apache/2.2.16 (Unix) mod_ssl/2.2.16 OpenSSL/1.0.0a PHP/5.3.3"
"83.169.2.20","200","Foobar"
"85.24.189.121","200","lighttpd/1.4.28"
"85.31.186.67","200","Apache/2.2.16 (Debian)"
"85.31.186.70","200","WSGIServer/0.1 Python/2.6.6"
"88.80.202.184","200","Apache/2.2.9 (Debian) DAV/2 PHP/5.2.6-1+lenny13 with Suhosin-Patch
   mod_python/3.3.1 Python/2.5.2 mod_ssl/2.2.9 OpenSSL/0.9.8g"
"88.191.144.163","200","Apache"
"88.190.18.245","200","Apache"
"88.163.96.138","200","nginx"
"89.236.64.10","200","lighttpd/1.4.29"
"89.31.112.91","200","Apache/2.2.13 (Linux/SUSE)"
"94.102.52.211","403","Apache"
"91.121.6.118","200","Apache/2.2.17 (CentOS)"
"92.243.30.228","200","Apache/2.2.9 (Debian)"
"93.92.200.68","200","Apache/2.2.16 (Debian)"
"91.152.238.249","200","Iriz Network"
"94.23.29.78","200","blank"
"94.242.1.53","200","lighttpd/1.4.19"
"95.179.58.57","401","RomPager/4.07 UPnP/1.0"
"94.23.52.151","200","Apache/2.2.12 (Ubuntu)"
Scan started at 2011-12-13-00-56-25.051000 with 100 threads
Scan ended 2011-12-13-00-56-47.535000
```

13 Appendix 6 Clock skew measurement data

```
"7165.59,0.062,"Tue, 13 Dec 2011 02:14:34 GMT",146.52.96.160,httpd"
"1.668,0.531,"Tue, 13 Dec 2011 00:15:10 GMT",106.187.39.171,Apache/2.2.14 (Ubuntu)"
"1.574,0.312,"Tue, 13 Dec 2011 00:15:13 GMT",173.255.217.222,Apache/2.2.14 (Ubuntu)"
"1.559,0.125,"Tue, 13 Dec 2011 00:15:31 GMT",178.25.89.6,OmniSecure/3.0a5"
"-63459332129.4,0.171,"Tue, 1 Jan 0001 00:00:00 GMT",178.124.156.184,RomPager/4.07 UPnP/1.0"
"-63459332129.4,0.281,"Tue, 1 Jan 0001 00:00:00 GMT",178.169.68.27,blank"
"1.559,0.203,"Tue, 13 Dec 2011 00:15:32 GMT",184.107.206.46,nginx/1.0.9"
"-17967.441,0.328,"Mon, 12 Dec 2011 19:16:03 GMT",184.56.202.50,httpd"
"-63459332132.5,0.062,"Tue, 1 Jan 0001 00:00:00 GMT",188.134.2.118,blank"
"122.543,0.062,"Tue, 13 Dec 2011 00:17:36 GMT",188.134.81.69,nginx/0.8.54"
"0.528,0.016,"Tue, 13 Dec 2011 00:15:51 GMT",188.40.181.33,lighttpd/1.4.28"
"2.528,0.641,"Tue, 13 Dec 2011 00:15:53 GMT",192.138.213.236,Apache/2.2.9 (Fedora)"
"1.528,0.047,"Tue, 13 Dec 2011 00:15:53 GMT",193.41.7.64,Apache/2.2.16 (Debian)"
"1070.528,0.047,"Tue, 13 Dec 2011 00:33:42 GMT",194.14.172.60,lighttpd/1.4.26"
"1.528,0.125,"Tue, 13 Dec 2011 00:15:53 GMT",194.88.206.104,nginx/0.7.62"
"-63459332152.5,0.297,"Tue, 1 Jan 0001 00:00:00 GMT",2.132.135.23,RomPager/4.07 UPnP/1.0"
"1.512,0.406,"Tue, 13 Dec 2011 00:15:54 GMT",198.133.224.147,Apache/2.2.9 (Fedora)"
"-1.488,0.344,"Tue, 13 Dec 2011 00:15:52 GMT",206.253.164.169,nginx/1.0.5"
"1.512,0.172,"Tue, 13 Dec 2011 00:15:56 GMT",209.6.82.6,Apache"
"1.512,0.031,"Tue, 13 Dec 2011 00:15:57 GMT",212.12.40.162,Zope/(Zope 2.9.7-final, python 2.4.6, linux2) ZServer/1.1"
"1.512,0.156,"Tue, 13 Dec 2011 00:15:57 GMT",212.114.254.91,nginx"
"1.512,0.141,"Tue, 13 Dec 2011 00:15:58 GMT",212.232.42.95,lighttpd/1.4.29"
"1.496,0.047,"Tue, 13 Dec 2011 00:16:13 GMT",213.232.95.57,Apache/2.0.52 (CentOS)"
"-56.504,0.078,"Tue, 13 Dec 2011 00:15:15 GMT",217.147.161.13,nginx/0.7.65"
"1.496,0.125,"Tue, 13 Dec 2011 00:16:13 GMT",213.156.229.195,Apache/2.2.9 (Debian) PHP/5.2.6-1+lenny13 with Suhosin-Patch
  mod_ssl/2.2.9 OpenSSL/0.9.8g"
"1.496,0.593,"Tue, 13 Dec 2011 00:16:13 GMT",213.155.17.24,Apache/2.2.16"
"-230.504,1.047,"Tue, 13 Dec 2011 00:12:21 GMT",213.170.70.61,Apache/2.2.17 (Win32) PHP/5.3.5"
"1.496,0.125,"Tue, 13 Dec 2011 00:16:14 GMT",217.210.127.18,Apache"
"1.481,0.188,"Tue, 13 Dec 2011 00:16:16 GMT",31.207.192.133,lighttpd/1.4.26"
"1.481,0.032,"Tue, 13 Dec 2011 00:16:19 GMT",46.4.248.202,nginx/0.7.67"
"5.481,3.719,"Tue, 13 Dec 2011 00:16:22 GMT",46.249.16.11,nginx/0.8.53v
"1.465,0.062,"Tue, 13 Dec 2011 00:16:34 GMT",62.141.60.50,Apache/2.2.9"
"-195735347.535,0.172,"Thu, 29 Sep 2005 13:20:46 GMT",62.168.243.54,Apache/1.3.34 (Debian)"
"-18043.535,0.203,"Mon, 12 Dec 2011 19:15:50 GMT",66.108.39.87,httpd"
"489.449,0.187,"Tue, 13 Dec 2011 00:24:44 GMT",68.48.83.242,Apache/2.2.14 (Ubuntu)"
"1.449,0.406,"Tue, 13 Dec 2011 00:16:37 GMT",69.164.196.4,Apache/2.2.17 (Ubuntu)"
"-156.551,0.265,"Tue, 13 Dec 2011 00:14:02 GMT",71.239.91.32,nginx/1.0.5"
"1.449,0.312,"Tue, 13 Dec 2011 00:16:55 GMT",75.145.125.59,Apache/2.2.14 (Ubuntu)v
"1.434,0.266,"Tue, 13 Dec 2011 00:16:56 GMT",76.250.48.108,lighttpd/1.4.28v
"-1217.551,0.39,"Mon, 12 Dec 2011 23:56:37 GMT",75.145.125.61,Apache/2.2.21 (Unix) mod_ssl/2.2.21 OpenSSL/1.0.0d PHP/5.3.6"
"-63459332219.6,0.156,"Tue, 1 Jan 0001 00:00:00 GMT",77.37.172.127,blankv
"-139.582,0.109,"Tue, 13 Dec 2011 00:14:41 GMT",77.64.153.140,Apache/2.2.17 (Win32) mod_ssl/2.2.17 OpenSSL/0.9.8o
  PHP/5.3.4 mod_perl/2.0.4 Perl/v5.10.1"
"11.418,0.531,"Tue, 13 Dec 2011 00:17:27 GMT",78.231.24.103,Apache/2.2.14 (Ubuntu)"
"1.403,0.047,"Tue, 13 Dec 2011 00:17:18 GMT",78.46.47.67,Apache/2.2.9 (Debian) DAV/2 PHP/5.2.6-1+lenny13 with Suhosin-Patch
  mod_ssl/2.2.9 OpenSSL/0.9.8g mod_perl/2.0.4 Perl/v5.10.0"
"1.403,0.032,"Tue, 13 Dec 2011 00:17:19 GMT",78.47.156.38,Apache/2.2.16 (Debian)"
"1.403,0.313,"Tue, 13 Dec 2011 00:17:20 GMT",78.47.244.184,Apache"
"2.403,0.047,"Tue, 13 Dec 2011 00:17:22 GMT",78.86.34.19,httpd"
"0.371,0.093,"Tue, 13 Dec 2011 00:17:38 GMT",81.25.48.117,Apache/2.2.21 (Unix) mod_ssl/2.2.21 OpenSSL/1.0.0e DAV/2"
"2.371,0.015,"Tue, 13 Dec 2011 00:17:41 GMT",82.103.129.142,Apache"
"2.371,0.718,"Tue, 13 Dec 2011 00:17:46 GMT",83.169.2.20,Foobar"
"1.371,0.015,"Tue, 13 Dec 2011 00:17:46 GMT",83.169.5.75,Apache/2.2.8 (Ubuntu) mod_jk/1.2.25 mod_python/3.3.1
  Python/2.5.2 PHP/5.2.4-2ubuntu5.17 with Suhosin-Patch mod_ssl/2.2.8 OpenSSL/0.9.8g mod_perl/2.0.3 Perl/v5.8.8"
"40.356,0.141,"Tue, 13 Dec 2011 00:18:42 GMT",85.228.251.54,Apache/2.2.16 (Unix) mod_ssl/2.2.16 OpenSSL/1.0.0a PHP/5.3.3"
"1.356,0.235,"Tue, 13 Dec 2011 00:18:03 GMT",85.223.118.207,Apache/2.2.21 (FreeBSD) Phusion_Passenger/3.0.9 PHP/5.3.8 with Suhosin-Patch"
"-58.644,0.032,"Tue, 13 Dec 2011 00:17:04 GMT",85.24.189.121,lighttpd/1.4.28"
"1.34,0.062,"Tue, 13 Dec 2011 00:18:04 GMT",85.31.186.67,Apache/2.2.16 (Debian)"
"1.34,0.078,"Tue, 13 Dec 2011 00:18:04 GMT",85.31.186.70,WSGIServer/0.1 Python/2.6.6"
"2.324,0.468,"Tue, 13 Dec 2011 00:18:08 GMT",88.163.96.138,nginx"
"-4.676,0.031,"Tue, 13 Dec 2011 00:18:14 GMT",88.190.18.245,Apache"
"1.324,0.031,"Tue, 13 Dec 2011 00:18:21 GMT",88.191.144.163,Apache"
"1.324,0.015,"Tue, 13 Dec 2011 00:18:22 GMT",88.80.202.184,Apache/2.2.9 (Debian) DAV/2 PHP/5.2.6-1+lenny13 with Suhosin-Patch
  mod_python/3.3.1 Python/2.5.2 mod_ssl/2.2.9 OpenSSL/0.9.8g"
"13.309,0.125,"Tue, 13 Dec 2011 00:18:39 GMT",89.236.64.10,lighttpd/1.4.29"
"167.309,0.266,"Tue, 13 Dec 2011 00:21:14 GMT",89.31.112.91,Apache/2.2.13 (Linux/SUSE)"
"1.293,0.0,"Tue, 13 Dec 2011 00:18:45 GMT",91.121.6.118,Apache/2.2.17 (CentOS)"
"-1.707,0.109,"Tue, 13 Dec 2011 00:18:43 GMT",91.152.238.249,Iriz Network"
"1.278,0.032,"Tue, 13 Dec 2011 00:18:49 GMT",92.243.30.228,Apache/2.2.9 (Debian)"
"-53.738,0.063,"Tue, 13 Dec 2011 00:18:13 GMT",94.102.52.211,Apache"
"1.262,0.078,"Tue, 13 Dec 2011 00:19:08 GMT",93.92.200.68,Apache/2.2.16 (Debian)"
"1.231,0.016,"Tue, 13 Dec 2011 00:19:11 GMT",94.23.29.78,blank"
"1.231,1.063,"Tue, 13 Dec 2011 00:19:13 GMT",94.23.52.151,Apache/2.2.12 (Ubuntu)"
"-2.769,0.078,"Tue, 13 Dec 2011 00:19:20 GMT",94.242.1.53,lighttpd/1.4.19"
"-63459332368.8,0.219,"Tue, 1 Jan 0001 00:00:00 GMT",95.179.58.57,RomPager/4.07 UPnP/1.0"
1.231,3.578,"Tue, 13 Dec 2011 00:19:28 GMT",95.106.79.38,Apache/2.2.17 (Win32) PHP/5.3.6"
3.199,2.39,"Tue, 13 Dec 2011 00:19:55 GMT",mpaa.i2p,blank"
4.199,3.531,"Tue, 13 Dec 2011 00:19:55 GMT",forum.i2p,blank"
```

```
4.199,3.64,"Tue, 13 Dec 2011 00:19:55 GMT",freshcoffee.i2p,Apache"
"-0.801,4.828,"Tue, 13 Dec 2011 00:19:51 GMT",redzara.i2p,blank"
"3.059,2.453,"Tue, 13 Dec 2011 00:19:58 GMT",hq.postman.i2p,blank"
"3.059,2.953,"Tue, 13 Dec 2011 00:19:58 GMT",i2p-bt.postman.i2p,blank"
"3.199,6.843,"Tue, 13 Dec 2011 00:19:56 GMT",complication.i2p,blank"
"10.199,8.875,"Tue, 13 Dec 2011 00:20:01 GMT",ugha.i2p,blank"
"3.809,2.985,"Tue, 13 Dec 2011 00:20:01 GMT",www.i2p2.i2p,blank"
"2.809,2.36,"Tue, 13 Dec 2011 00:20:01 GMT",perv.i2p,blank"
"-25.238,3.109,"Tue, 13 Dec 2011 00:19:35 GMT",trac.i2p2.i2p,blank"
"6.059,7.75,"Tue, 13 Dec 2011 00:20:02 GMT",zzz.i2p,blank"
"3.809,6.031,"Tue, 13 Dec 2011 00:20:02 GMT",i2p-projekt.i2p,blank"
"2.762,4.672,"Tue, 13 Dec 2011 00:20:03 GMT",tracker2.postman.i2p,blank"
"13.059,11.344,"Tue, 13 Dec 2011 00:20:08 GMT",eepsites.i2p,blank"
"7.762,5.188,"Tue, 13 Dec 2011 00:20:09 GMT",i2pbote.i2p,blank"
"4.199,15.14,"Tue, 13 Dec 2011 00:19:56 GMT",bl.i2p,blank"
"2.637,1.703,"Tue, 13 Dec 2011 00:20:08 GMT",update.killyourtv.i2p,blank"
"3.653,4.329,"Tue, 13 Dec 2011 00:20:08 GMT",diftracker.i2p,blank"
"2.637,5.25,"Tue, 13 Dec 2011 00:20:07 GMT",planet.i2p,blank"
"20.199,18.296,"Tue, 13 Dec 2011 00:20:12 GMT",riaa.i2p,blank"
"6.637,8.125,"Tue, 13 Dec 2011 00:20:11 GMT",pastethis.i2p,blank"
"16.762,15.375,"Tue, 13 Dec 2011 00:20:17 GMT",docs.i2p2.i2p,blank"
```

14 Appendix 7 Country distribution of I2P neighbors (IP set reproduction part)

Russian Federation	115
United States	68
Germany	58
France	43
Ukraine	15
Sweden	14
Canada	11
United Kingdom	9
Finland	9
Netherlands	8
Spain	5
Japan	3
Switzerland	3
Portugal	2
Italy	2
Israel	2
Belarus	2
Bulgaria	2
Austria	2
Uruguay	1
Tunisia	1
Slovenia	1
Saudi Arabia	1
Serbia	1
Poland	1
Norway	1
Mexico	1
Luxembourg	1
Kazakhstan	1
Republic of Korea	1
India	1
Ireland	1
Hungary	1
Gambia	1
Denmark	1
Czech Republic	1
Costa Rica	1
Colombia	1
Brazil	1
Belgium	1
Australia	1

15 Appendix 8 Country distribution of I2P neighbors (IP set improvement part)

Russian Federation	3456	Turkey	8
United States	1378	Hungary	8
Germany	882	Bulgaria	8
France	839	Saudi Arabia	7
Ukraine	380	Lithuania	7
United Kingdom	259	Iran, Islamic Republic	7
Canada	187	Uruguay	6
Finland	156	United Arab Emirates	6
Italy	155	Venezuela	5
Sweden	129	Korea, Republic of	5
Australia	90	Costa Rica	5
Brazil	82	Slovenia	4
Spain	76	Luxembourg	4
Belarus	75	Jordan	4
Mexico	69	Armenia	4
New Zealand	68	Trinidad and Tobago	3
Netherlands	66	Tunisia	3
China	66	Philippines	3
Austria	64	Jersey	3
Japan	55	Iceland	3
Kazakhstan	38	Indonesia	3
Switzerland	38	Cyprus	3
Poland	36	Bosnia and Herzegovina	3
Belgium	34	Azerbaijan	3
India	31	Uzbekistan	2
Colombia	31	Serbia	2
Norway	28	Puerto Rico	2
Israel	28	Panama	2
Portugal	26	Malta	2
Denmark	21	Kyrgyzstan	2
Czech Republic	21	Georgia	2
Argentina	20	Ecuador	2
Ireland	19	Dominican Republic	2
Thailand	17	Bolivia	2
Latvia	17	Albania	2
Anonymous Proxy	16	Turkmenistan	1
Chile	15	Togo	1
Taiwan	12	Singapore	1
Greece	12	Seychelles	1
Slovakia	11	Pakistan	1
Peru	11	Oman	1
Croatia	11	Mozambique	1
Estonia	11	Montenegro	1
South Africa	10	Kuwait	1
Romania	10	Kenya	1
Hong Kong	10	Address not found	1
Syrian Arab Republic	9	Guam	1
Malaysia	9	Ethiopia	1
Moldova, Republic of	9	Egypt	1
Algeria	9	Cuba	1
Vietnam	8		

Index