University of Amsterdam

System and Network Engineering

Research Project 2

Crawling the USENET for DMCA

Author:
Eddie Bijnen, student

June 19, 2015

Abstract

To combat copyright infringement copyright holders have the ability to request that files are to be removed from USENET. However, there was no data available which files are being removed from USENET or how fast the data is being removed.

In this report and subsequent website we have shed light on this DMCA activity. We have gathered information on availability, time till removed, speed of checking for available articles, USENET article growth patterns, trends in file naming and correlation of unavailable articles per provider. We have found that the behavior of DMCA between USENET providers differs greatly. And that the behavior of USENET reseller may differ from the parent company.

Contents

1 Introduction

USENET has been around since the early days of the Internet. A place for discussions, questions and news articles. The use of a client server model gives the reader access to a huge amount of information at high speed. These characteristics have made USENET very popular in the pirating scene. In an attempt to combat the loss of income to the copyright holders, privately funded organisations have been hired to start sending removal requests. The USENET providers have been swamped under the mountain of automated requests and have given the copyright holders direct access in order to comply with local law. This allows the copyright holders to remove content without oversight and completely automatically.
During the research period we attempted to devise a method of creating a reliable index of the original state of the USENET network and to create an user-friendly interface to display which files have been removed from which provider.

1.1 Research question

1. *Can a comprehensive database, including DCMA take downs, be created?*
2. *What kinds of method exist to keep article availability up to date?*
3. *Is it feasible to keep the entire USENET article availability up to date?*

1.2 Related work

1.2.1 nZEDb

nZEDb is an open source project that crawls the USENET for subjects and attempts to match it to a movie, series or piece of software and load additional information from several sources in order to create a website with detailed information about what is available on the Internet. However there is no ability to check availability of what has been indexed.

1.3 Approach

The first step is to identify how an USENET provider responds when a removed article is requested. Is an error code returned?; Are empty files provided?; Is the header index updated to identify that the article has been removed? Once this information is gained, a database can be created with all header information of each newsgroup. This database has an collection of all information that has been available on the USENET. We are then able to continue to check the availability of the articles for multiple USENET providers.

2 Background

NNTP stands for Network News Transfer Protocol, originally specified in 1986 in RFC977. NNTP was immensely popular during the earlier years of the Internet. However, with the coming of the web, forums and social media, the original purpose of the USENET, which was to discuss and publish information, has declined. A new type of user has discovered the USENET: the Internet pirates. The ability to store files on a fast, central server that others can access anonymously has made it an ideal place to store legitimate and illegitimate files.

2.1 Digital Millennium Copyright Act

When a copyright holder believes that their files have been placed on-line without there permission. They can make use of the US law DMCA or the European Copyright Directive. Both laws have protection for hosting providers and Internet providers that they can not be held liable for files uploaded by there users. As long as they are unaware of the copyrighted material. An take down notice is a notice to the provider that copyrighted material is on there infrastructure that the copyright holder has not approved. To comply with the copyright laws these files need to be made unavailable. An take down notice must contain the following [2] [10] [1] :

1. Clear identification of the person or entity submitting the DMCA Notice.
2. Clearly stated relationship to the copyright holder (self or authorized agent).
3. Message-IDs for all articles the DMCA Notice is requesting to take down.
4. Clear statement, that the information in the notification is accurate and that you are copyright holder, or authorized to act on behalf of the copyright holder.
5. A "physical or electronic signature" of an authorized person to act on behalf of the owner.

There is no set amount of time in which a USENET provider needs to comply with a take down notice. However it should be within a reasonable amount of time. The general consensus is that DMCA take down should be handled within days.

2.2 The NNTP Protocol

NNTP, like many of the protocols of that time, is a stream-based connection very similar to HTTP and SMTP. It connects to a central server that stores articles to be retrieved by the client. These articles are divided into newsgroups; each newsgroup has its own subject. One can join a group by typing *GROUP %groupname.%* Upon entering a group the start article and end article id that are available on the server are returned. A list of available articles and their subjects can be retrieved with the command *XOVER %start article% - %end article%.* The individual article can be retrieved by issuing the command

ARTICLE %article ID.% Table 1 show an overview of common used commands

NNTP uses command codes to identify server responses to given commands. The first three characters identify the command, followed by a parameter in some cases. Commands codes are categorized as follows:

```
The first digit of the response broadly indicates the success, failure,
or progress of the previous command:
      1xx - Informative message
      2xx - Command completed OK
      3xx - Command OK so far; send the rest of it
      4xx - Command was syntactically correct but failed for some reason
      5xx - Command unknown, unsupported, unavailable, or syntax error

The next digit in the code indicates the function response category:

      x0x - Connection, setup, and miscellaneous messages
      x1x - Newsgroup selection
      x2x - Article selection
      x3x - Distribution functions
      x4x - Posting
      x8x - Reserved for authentication and privacy extensions
      x9x - Reserved for private use (non-standard extensions)
```

Table 1: Common client commands

Code	**Command**	**Parameter**
ARTICLE	Retrieve Article	Message ID or server article number.
HEAD	Retrieve Article Header	Message ID or server article number.
STAT	Retrieve Article Statistics	Server article number
GROUP	Select Newsgroup	Newsgroup name
LIST	List Newsgroups	N/A
XOVER	Retrieve Subjects posted to that newsgroup	Range of server article numbers

An example of a typical NNTP conversation:

```
CLIENT: telnet news.sunnyusenet.com 119
CLIENT: Trying 85.12.14.42...
CLIENT: Connected to news.sunnyusenet.com.
CLIENT: Escape character is ’^]’.
SERVER: 200 news.sunnyusenet.com NNRP Service Ready
CLIENT: AUTHINFO USER %username%
SERVER: 381 PASS required
CLIENT: AUTHINFO PASS %password%
SERVER: 281 news.sunnyusenet.com NNRP Service Ready
CLIENT: GROUP alt.binaries.boneless
```

```
SERVER: 211 6974665886 7319575963 14294241848 alt.binaries...
CLIENT: XOVER 14294241847-14294241848
SERVER: 224 Overview Information Follows
SERVER: 14294241847 [Art-of-Use.net] - [050/110] - "XN0YPT...
SERVER: 14294241848 xxx - [057/457] - "LQOcCXzgQBwbre0oizt...
SERVER: .
SERVER:
CLIENT: HEAD 14294241847
SERVER: 221 14294241847 <Part7of274.D873C0B04A81426BB7F55E...
SERVER: Path: not-for-mail
SERVER: From: yEncBin@Poster.com (yEncBin)
SERVER: Sender: yEncBin@Poster.com
SERVER: Newsgroups: alt.binaries.art-of-usenet,alt.binarie...
SERVER: Subject: [Art-of-Use.net] - [050/110] - "XN0YPTIUH...
SERVER: X-Newsposter: yEncBin Poster v1.0.343 (http://memb...
SERVER: Message-ID: <Part7of274.D873C0B04A81426BB7F55ECB30...
SERVER: Date: 24 Mar 2015 12:55:28 GMT
SERVER: Lines: 3063
SERVER: Organization: bullcat
SERVER: X-Received-Body-CRC: 501729876
SERVER: Bytes: 398897
SERVER: X-Original-Bytes: 398768
SERVER: X-Received-Bytes: 399010
SERVER: .
```

2.3 NNTP Architecture

Posting an article is done by posting it to one's own USENET provider. The USENET provider accepts the article, stores it, and marks it to be send to its neighbors. Each provider has an designated queue for each of its neighboring providers. It notifies its neighbor with the *IHAVE* command. The receiving provider can choose three options: ”335 Send article to be transferred”, ”435 Article not wanted” or ”436 Transfer not possible; try again later”. To which server the connections are setup is up to the administrators of the USENET providers. An outdated map of the USENET landscape can be found in appendix B on page 33. Its information could, however, not be verified [9].

2.4 Definitions

Header	The head information of an article
Header index	The output of LIST command
Article	The complete article
Article-id	Unique ID for that article
Article number	The number of that article in a specific newsgroup
Part	The name nZEDb has used for an article
File	An file existing of one or more articles
File set	An fileset exists of one or more files
Collection	The name nZEDb has used for a fileset
Newsgroup	A container of articles with a specific subject
DCMA	Stands for Digital Millennium Copyright Act
NNTP	Network News Transfer Protocol
RAR	A compression and split standard
PAR	A parity archive with the ability to detect and repair corrupted files
Backfill	retrieving header index from a previous period

2.5 File Encoding

NNTP does not have file support built-in. Because of this shortcoming files need to placed inside the article. Because NNTP is limited to 8-bit extended ASCII, files need to be encoded and encapsulated. The common way to do this is to use yEnc, this has a small overhead between 1 and 2% [8]. yEnc attachments can be identified by "=ybegin" and "=yend"

2.6 File Structure

Because of the way file attachments have been hacked into the NNTP protocol, tricks need to be performed to upload large files. The typical way to do this is with RAR and PAR [3]. Figure 1 illustrates this procedure. A large file is compressed and split up using RAR. PAR is used to make these files error resistant by adding parity to the files. These files get encoded with yEnc and split over multiple articles. Because of the limitation of the amount of lines a single article can contain, the number of articles is significantly higher than the number of files being uploaded. The limit on the number of lines is defined by the USENET provider who will return a "441 Article too big for this server" if the server limit is exceeded when posting. The entire set of uploaded files will be referred to as fileset

Table 2: File count of an typical TV episode of 1.1GiB

File Type	Number of files
Orginal File	1
Compressed files	21
Compressed files & parity	28
Number of articles on USENET	1605

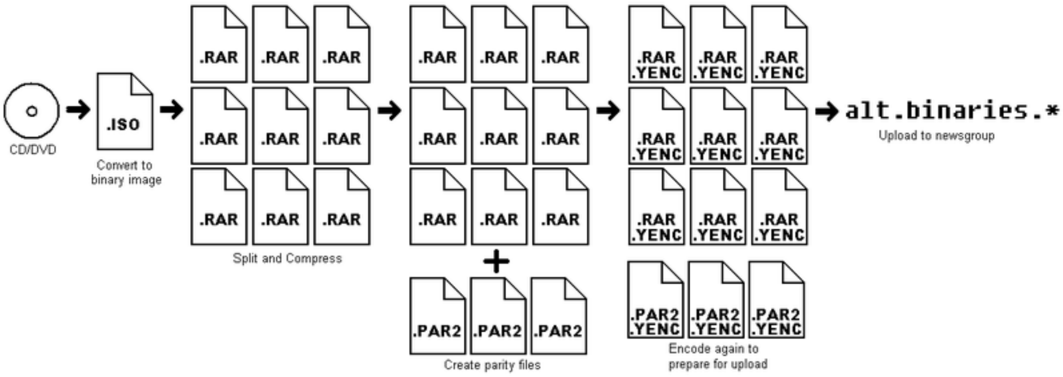

Figure 1: Posting of large binaries

2.7 Providers

There are a lot of companies selling USENET access, however, due to the huge amount of storage needed, few of these companies have their own storage. Almost all companies that sell USENET access resell a package from a larger provider. There are currently five large providers that provide USENET access who include binary groups. A complete list of USENET providers and their resellers compiled from Internet sources [11] [5] [6] can be found in Appendix C.

3 Methodology

To find out which files are being removed of the USENET we must first know which files are being posted to the USENET. To achieve this we store the output of the XOVER command which contains the subject, poster, date, newsgroup & article id. With this information we can retrieve articles or check their availability. For each check we store the fileset identifier, provider, date, if found. In Chapter 4.1 and 4.2 we explain how this is done.

3.1 Choosing providers

The choice of provider is primarily based on whether the provider provides an option for a trial account or not, and which accounts were available to us. We have attempted to maintain a mix of providers with different parent companies, shown in Table 3 as well as simultaneous connections as the retention in days.

Table 3: USENET Providers

Provider	**Owner**	**Trial/Paid**	**Conn**	**Retention**
Astraweb	Astraweb	Paid	50	2406
UseNeXT	Aviteo Ltd	14days/300GB	30	2013
Nextgen news	Nextgen news	4GB	30	unknown
Eweka	UNS Holdings	7days/10GB	8	2418
Fast Usenet	UNS Holdings	14days/15GB	40	2420
Giganews	UNS Holdings	14days/10GB	50	2367
Sunny Usenet	UNS Holdings	Paid	20	900
UNS	UNS Holdings	14days/10GB	10	2421
Hitnews	XENNEWS/RSP	Paid	20	1100
Bulknews	XSNews	10GB	30	900

3.2 Data structure

Our database exists of 3 main tables. 'Collections' where the file set information is stored, 'Availability' which contains the checks executed and there result. And 'Parts' which are the individual articles which are required for checking of availability. As well as the creation of NZB files that are used by download programs to initiate download of the fileset. An overview of tables in the database can be seen in Figure 2.

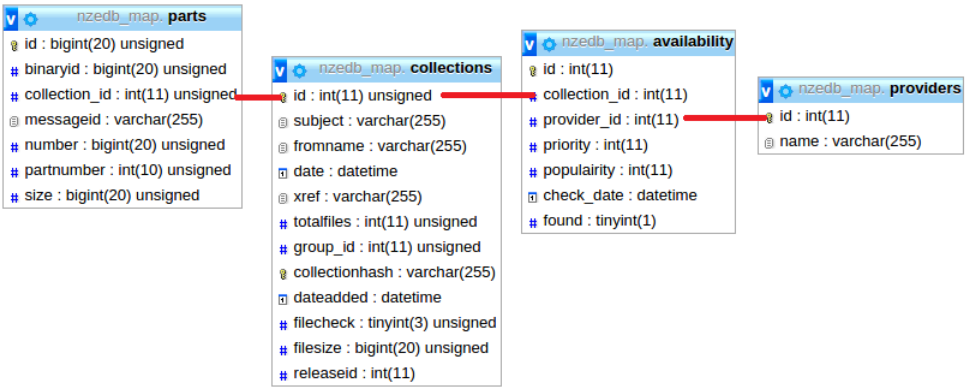

Figure 2: Primary MySQL Tables and relations

3.3 Performance

Performance of the crawler is a top priority as it is necessary to check hundreds of millions of articles. To be able to scan as many articles as possible the program has support for multithreading and batch jobs. To further increase speed we have used a python line profiling tool called kernprof to locate expensive commands and unnecessary waits. Furthermore, we have changed several SQL queries to avoid joins and server-side sorting. It is for instance a lot faster to request all records and have python select one at random than to request a single random record via SQL. An indepth profile of the main functions can be found in Appendix A. A summery of the main functions of the USENET crawler and how much time is spent in each function can be found in table 4.

Table 4: Time spent on action

Task	When	Average time spent
Priority tasklist	Once every 5min	6.344ms
Most outdated tasklist	Once every 50.000 articles	4.941ms
Get article-ID	For each article	1ms
Checking one article	For each article	174ms

3.4 Infrastructure

The infrastructure started out initially with a single Dell PowerEdge R210 with a single disk. During the testing of the feasibility of the proposal it turned out that this would insufficient diskspace and too limited IOPS. A secondary server was add to provide storage and run SQL. Running both ZFS and SQL on a single server with limited amount of memory created more performance problems leading to a three-server structure.

Table 5: Servers

Name	**Amsterdam**	**ZFS**	**DB**
Purpose	Crawler	Storage	Database
CPU	4x1.87Ghz	4x2Ghz	8x2.5Ghz
RAM	8GB	8GB	16GB
Disk	2x500GB	4x2TB 1x500GB 1x60GB (SSD)	160GB
Software	Homemade & nZEDb	ZFS & NFS	MariaDB 10.0.17

3.5 Choice of software

3.5.1 Python

Python has been used to create the crawler, website and tools for retrieving statistics. The choice for using Python for the majority of the homemade software was primarily based on the fact that the researchers were proficient in the language. And that there is wide support and code snippets as well as a wide variety of libraries.

3.5.2 nZEDb

nZEDb is a open source project that aims to create a website where one can browse software/movies/TV series that are available for download from the USENET. For our project we have used a small part of the newsgroup-to-MySQL parser to create the initial index of newsgroups. The reason for using nZEDb and not creating a USENET parser ourselves is that it suits our needs, and has been widely tested.

3.5.3 ZFS

ZFS is being used as file system for the MySQL database. The choice for ZFS was made because of its ability to compress data and store it in multiple storage tiers. The use of multiple tiers has meant that we can make efficient use of the fast but relative small SSD and still be able to store a very large database.

3.5.4 MariaDB

The choice of MySQL engine was based on the benchmarks[4] and is recommended by the nZEDb development team[7], as well as being compatible with phpmyadmin and python.

4 Results

Over the course four months we have collected what is being posted to the USENET. During a twelve day period from 27th of March to 7th of May. We have tracked the availability of the filesets collected in the previous months.

4.1 Building a comprehensive database including DCMA takedowns

To understand what is being removed we need to have a complete list of material that is being posted to USENET. To achieve this we have made use of an existing piece of software called nZEDb. It is a piece of open-source software that receives the subject list from each newsgroup. It parses the subject field and is able to create sets of articles belonging to a file and fileset out of files. It does this through a specialized PHP extension that is written in C++ to gain the best performance. The results are three tables for parts, files and filesets. It saves the article number of the last received article so that it can request all new articles every five minutes by using the XOVER command receiving only the newly posted.
The results show that it takes hours for articles to be removed from the USENET provider that we use to index. It then takes time for the header index to be updated. We have been unable to determine how long it takes for a removed article to be removed from the header index. We have, however, observed removed articles that are still available in the header index. And we have observed headers being removed from the header index. By checking on a five minute interval the chance that an article has been indexed by other crawlers, identified as copyrighted material, requested for takedown, processed by the USENET provider and updated the header index is small. We are confident that we have a complete list of the articles that are being posted to the USENET.

4.2 What methods exist to keep article availability up-to-date?

There are five methods of checking article availability: ARTICLE, BODY, HEAD, STAT and XOVER. A quick recap:

Table 6: Common client commands

Command	**Command function**	**Parameter**
ARTICLE	Retrieve Article	Message ID or article number.
BODY	Retrieve Article Body	Message ID or article number.
HEAD	Retrieve Article Header	Message ID or article number.
STAT	Retrieve Article Statistics	Message ID or article number.
XOVER	Subjects posted to that newsgroup	Range of article numbers

4.2.1 ARTICLE & BODY command

A combination of both would be the ideal choice for checking availability as downloading the actual article makes sure it is available. The downside of this is that downloading the whole article instead of just the body results in far more data to transfer. And it could have legal implications as you are downloading millions of articles, some of which may contain illegal or copyrighted material.

4.2.2 XOVER command

The purpose of the XOVER command is to generate a list of article subjects the so called header index. However, while looking into this method it was discovered that some articles will still return a subject while it is no longer available to download. It is very likely that the list of subjects is only updated every now and again and does not return accurate availability information.

4.2.3 HEAD command

While the HEAD command gives acurate availability information it returns about ten lines of the article. This information is completely discarded, the STAT command returns even less data; it is there for faster response and requires less bandwidth.

4.2.4 STAT command

The STAT command is used to check the availability. The command STAT message-id has two possible results: ”223 0 message-id Article exists” — ”430 No article with that message-id” as specified in RFC3977. The server is not allowed to return an article if it can not produce the whole article as stated in RFC3977 section 6.2:

> RFC3977: 6.2. Retrieval of Articles and Article Sections
> The ARTICLE, BODY, HEAD, and STAT commands are very similar. They differ only in the parts of the article that are presented to the client and in the successful response code. The ARTICLE command is described here in full, while the other three commands are described in terms of the differences. As specified in Section 3.6, an article consists of two parts: the article headers and the article body.
> When responding to one of these commands, the server MUST present the entire article or appropriate part and MUST NOT attempt to alter or translate it in any way.

4.3 Verifying results

To verify that the results that are returned by the program are valid, we have run several checks to verify the consistency of the gathered data. There are three steps that need to be verified to determine the preciseness of the results.

1. Is the STAT command consistent?

2. How many articles do you need to check before a conclusion can be made about the file set?
3. Is the returned data consistent with users experience when downloading?

4.3.1 Is the STAT command consistent?

We have created a set of 10,000 articles from 10,000 filesets. This set we checked multiple times and stored the result. We then compared the results to discover a bug in the program: if the server would be slow to respond, articles would arrive out of order. After making the necessary alterations we came to the following numbers:

Number of articles in file set: 10,000
Number of deviating returns: 3
Chance that a file is indexed incorrectly 0.03%

The theory behind this deviation is that perhaps a file is removed between scans or a loadbalancer has routed the request to a server that might have the article cached where previously it was routed to a server that had not. None the less a deviation of 0.03% exists. We did not further investigate the cause of this deviation.

4.3.2 How many articles do you need to check before a conclusion can be made about the file set?

For performance reasons only a limited number of articles are checked per fileset. To investigate how many articles are required to make an accurate estimation of the availability of the fileset we have created a set of roughly 100.000 articles for 200 filesets. For these 200 filesets all articles were requested and both negative and positive replies for article availability were returned. However, there was zero deviation between the availability of a single article and the filesets it belonged to. The test was repeated and manual deviations were added to the results to verify that indeed the scripts and results were consistent.

The result meant that by checking a single article a very strong conclusion can be made about the availability of the entire filesets. Increasing the number of articles to check to two articles would decrease performance with 50%, resulting in the time required for scanning the same set would double. As fileset are checked at regular intervals any incorrect data would be visible and corrected.

4.3.3 Is the returned data consistent with user experience?

The files that we see that are removed are for the most part expected with user experience. We can see that popular illegally distributed TV series like "Arrow", "Game of Thrones" and "Better Call Saul" are available for a short time and after they are removed by the majority of the providers. But the majority of the files stays untouched.

4.4 Is it feasible to keep the entire USENET article availability up-to-date?

The feasibility of checking the entire USENET depends on how many articles exist, how large the file set is and the speed of the checking. We have computed multiple scenarios below.

4.4.1 Speed of availability checking

There are three factors that influence the speed at which articles can be checked: network latency, server latency and number of connections. Using our crawler we are able to practcally achieve between the 60K and 1M article checks an hour. For the network latency we have taken an average of five round trip times. With some interesting results, Network latency seems to have a limited effect especially when coupled with more connections. Which makes sense because the more questions can be asked, the less the round trip time affects checking time. However, server latency has an even larger impact; at Bulknews we can check double the amount of articles of UseNEXT with double the latency.

Table 7: Speed of checking availability per provider

Provider	**Articles checked in an hour**	**RTT**	**connections**	**Articles per connection**
Astraweb	658452	0.724ms	49	13438
Bulknews	1035587	14.057ms	30	34520
Eweka	184167	1.852ms	8	23021
Fast Usenet	229459	101.190ms	40	5736
Giganews	810995	6.580ms	49	16551
Hitnews	264489	15.967ms	19	13920
Nextgen news	59493	3.672ms	30	1983
Sunny Usenet	185594	1.079ms	10	18559
UNS	185594	1.473ms	10	18559
UseNeXT	554738	6.706ms	30	18491

4.4.2 Size of the USENET

The size of the USENET is measured in number of articles. There is no command to retrieve a count of all the articles on the server. There is, however, a command to retrieve the list of available newsgroups. The LIST command returns the name, reported high water mark, reported low water mark and the status of the group.

> RFC3977: 6.1.1.2 GROUP Description
> The successful selection response will return the article numbers of the first and last articles in the group at the moment of selection (these numbers are

> referred to as the ”reported low water mark” and the ”reported high water mark”) and an estimate of the number of articles in the group currently available.
>
> If the group is not empty, the estimate MUST be at least the actual number of articles available and MUST be no greater than one more than the difference between the reported low and high water marks. (Some implementations will actually count the number of articles).

By retrieving the first (newest) article number we can determine how many articles have been posted in a certain amount of time. If we take all available newsgroups and add their article count we know how many articles have been posted in those newsgroups. We can also tell by how much the number has incremented in a period of time. Adding all article counts of all available newsgroups we reach the number of five quadrillion articles ever posted to USENET in the groups that still exist today.

However, many of these articles no longer exist and as explained in ”4.3 Verifying results” we do not need to check every single article to make an assumption about a fileset. As we cannot get data about filesets this number does not help us much.

4.4.3 Number of filesets

As the number of file sets can not be reliably gathered after it has been posted as explained in Section 4.1. We use the reliable data that we have gathered to do an extrapolation.

We have collected four months of reliable fileset data for the alt.binaries.boneless group, the largest news group on USENET. In 122 days 2.268.165 file sets have been added to the group ’boneless’. The group ’boneless’ is responsible for 11.79% of all posted articles. If we assume that other groups have the same amount of articles in a fileset, we can make the following calculation.
The following table and graphics rely on weak data. The following assumptions had to be made to make this calculation:

1. The average number of articles in a fileset is the same for all newsgroups.
2. The number of articles posted between 1-04-2015 and 14-04-2015 are representative for an average year.
3. The growth of the amount of articles is stable.

Table 8: Estimade number of filesets published after x days

Group	percentage	Number of days	filesets
alt.binaries.boneless	11.79%	1	18591
alt.binaries.boneless	11.79%	122	2.268.165
USENET	100%	1	157684
USENET	100%	10	1576840
USENET	100%	50	7884200
USENET	100%	100	15768400
USENET	100%	500	78842000
USENET	100%	1000	157684000
USENET	100%	2000	315368000
USENET	100%	2500	394210000

4.4.4 Growth of the USENET

Between the dates 1-04-2015 and 14-04-2015 we have counted how many articles have been uploaded to the USENET per newsgroup.

Table 9: USENET Growth

Group	Articles added	Percentage of the total
USENET	1033748563	100.00%
alt.binaries.boneless	121872988	11.79%
alt.binaries.mom	49293427	4.77%
alt.binaries.dvd	44495265	4.30%
alt.binaries.nl	43648904	4.22%
alt.binaries.hdtv	43056247	4.17%
alt.binaries.bloaf	41553241	4.02%
alt.binaries.cores	39590313	3.83%
alt.binaries.u-4all	35680191	3.45%
alt.binaries.test	35261418	3.41%
alt.binaries.erotica	30740033	2.97%

By taking the number of file sets and dividing them by the speed of which file sets can be checked, the following table can be created. This table shows how many days it takes to check all file sets that are posted in x days. This is assuming a single account. Having multiple account to do checking from would speed up this process. The speed of which articles are checked can be found in Table 7

Table 10: Number of days required to check all file sets posted

Days	Astraweb	Bulknews	Eweka	Fastnews	Giganews
1	0.01	0.01	0.04	0.03	0.01
10	0.1	0.06	0.36	0.29	0.08
50	0.5	0.32	1.78	1.43	0.41
100	1	0.63	3.57	2.86	0.81
500	4.99	3.17	17.84	14.32	4.05
1000	9.98	6.34	35.68	28.63	8.1
2000	19.96	12.69	71.35	57.27	16.2
2500	24.95	15.86	89.19	71.58	20.25

Days	Hitnews	Nextgen	Sunny news	UNS	UseNext
1	0.02	0.11	0.04	0.04	0.01
10	0.25	1.1	0.35	0.35	0.12
50	1.24	5.52	1.77	1.77	0.59
100	2.48	11.04	3.54	3.54	1.18
500	12.42	55.22	17.7	17.7	5.92
1000	24.84	110.44	35.4	35.4	11.84
2000	49.68	220.87	70.8	70.8	23.69
2500	62.1	276.09	88.5	88.5	29.61

This crawling method treats all file sets equally; in a search provider situation you would take popularity of file sets into account. Items that are no longer available are still checked in this calculation. Changing this behavior would decrease the number of days. The exact amount would depend on which percentage of the files still exists.

4.5 Interpreting gathered data

We have started collecting fileset information for alt.binaries.boneless on the first of December 2014 adding the top 10 newsgroups at a later time. Checking the availability for articles began on the 28th of March 2015 and ended on 7th of April 2015. During this time 3,211,532 file sets and 382,364,225 availability checks have been collected.

4.5.1 Cleaning data

Availability checks contain a lot of data, however, there are only four interesting records. The first and last available and the first and last unavailable. But before we can carve these records out of the database. We need to account for inconsistent results as explained in 4.3.1. An error rate of 3 in 10,000 is not significant however when deduplicating records the measurement errors are significantly increased compared to legitimate data. To counter act the measurement errors an script will compare each record with it's predecessor and successor. If the predecessor and successor are the same but the record deviates it is corrected.

To make the data manageable we need to deduplicate the availability records. One to create the lowest date and a unique collection_id Provider_id and found status, the second to create a table with the latest records with a unique collection_id Provider_id and found status. Combined these two queries contain the interesting availability data for all file sets.

```
CREATE TABLE availability_dedup_MIN AS
SELECT ‘id‘, ‘collection_id‘, ‘provider_id‘, ‘priority‘, ‘populairity‘,
MIN(‘check_date‘), ‘found‘
FROM availability
GROUP BY ‘collection_id‘,‘provider_id‘, ‘found‘;
```

```
CREATE TABLE availability_dedup_MAX AS
SELECT ‘id‘, ‘collection_id‘, ‘provider_id‘, ‘priority‘, ‘populairity‘,
MAX(‘check_date‘), ‘found‘
FROM availability
GROUP BY ‘collection_id‘,‘provider_id‘, ‘found‘;
```

To get a collection of all unavailable data containing the first unavailable record and the last available record for file sets that are no longer available, a third subset was made:

```
INSERT INTO availability_unavail
SELECT * FROM availability_dedup_MIN
WHERE ‘found‘ = 0;

INSERT IGNORE INTO availability_unavail
SELECT * FROM availability_dedup_MAX
WHERE ‘collection_id‘
IN (SELECT ‘collection_id‘ FROM ‘availability_unavail‘)
AND ‘found‘ = 1;
```

For speed purposes the following table was created. It shows a subset of collections that are no longer available on at least one provider.

```
INSERT IGNORE INTO availablity_collections
SELECT * FROM collections
WHERE ‘id‘ IN
(SELECT ‘collection_id‘ FROM ‘availability_unavail‘);
```

Query for the creation of the EFNet subset. This query contains the latest records where the subject name contains ”a.b.teevee@EFNet”. The ”a.b.teevee@EFNet” tag shows it is a release of a specific pirate release group known for post TV series. This dataset only contains the last found and last non-found records and can therefore not be used to calculate the time it took for these files to be removed.

```
INSERT INTO availability_EFNET SELECT * FROM availability_dedup_MAX
WHERE ‘collection_id‘ IN
(SELECT id FROM collections WHERE subject LIKE ’%a.b.teevee@EFNet%’);
```

Table 11: MySQL tables and content

Table name	Rows	Containing
availability	382,364,225	All availability checks
collections	3,211,532	All filesets
availability_dedup_MAX	36,337,715	The last dated found and unfound check
availability_dedup_MIN	37,272,000	The first dated found and unfound check
availability_unavail	1,215,301	The first unfound and last found check
availability_EFNET	538,538	The last dated found and unfound for file sets containing: a.b.teevee@EFNet (Not suited for time calculations)
availability_collections	109,358	Filesets with unfound checks

4.5.2 Completion per provider

To calculate the percentage of available filesets per provider we counted the number of available and unavailable records in the availability_dedup_MAX table. The table only contains one record per provider for an available file set and one record per provider for an unavailable file set. By counting the number of available records and dividing it by the total, we can calculate the percentage of available file sets. The overall availability is shown in Figure 4.5.2

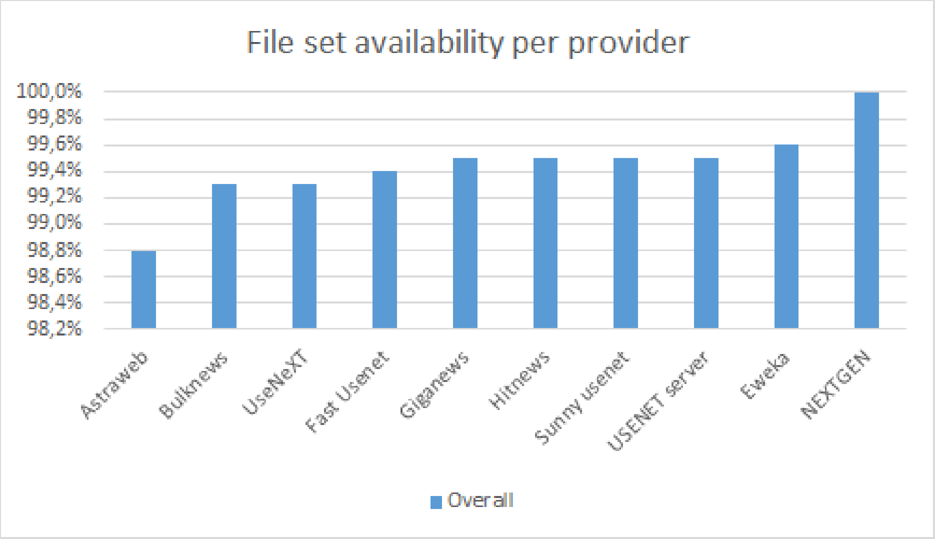

Figure 3: File set availability per provider

4.5.3 Completion for TV series subset

Indexing which data is copyrighted is outside the scope of the project. However, there is a very clear structure for the release of TV episodes. The release group that is responsible for the majority of the released TV shows is known as a.b.teevee@EFNet. They use this prefix in all their postings to the USENET. This makes it very easy to check the completeness of their releases per provider. The EFNet availability is shown in Figure 4

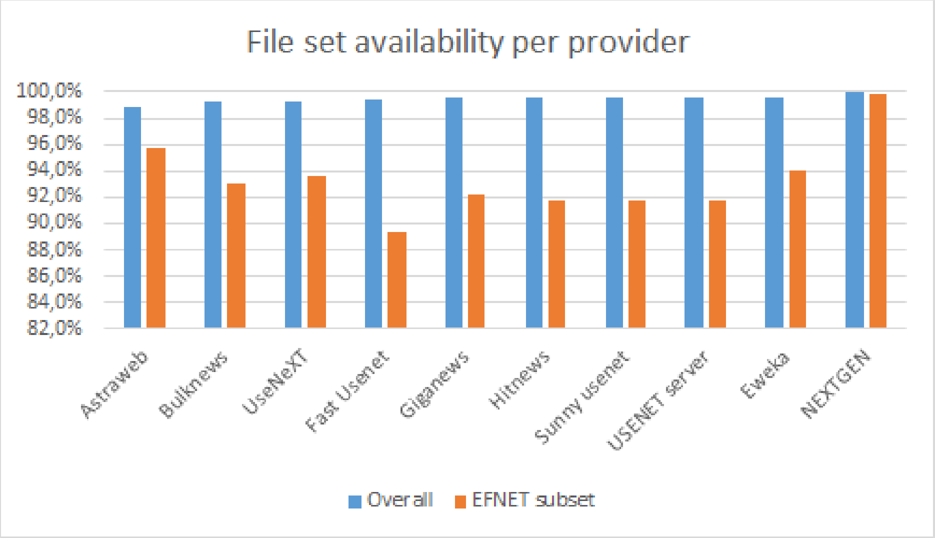

Figure 4: EFNet File set availability per provider

4.5.4 Posting behavior

Out of the gathered date we can determine at which dates and times articles are posted to the USENET. This data is based on the header information of the first article of each file set. We have gathered four weeks of data. The graph in Figure 5 shows an average of these four weeks.

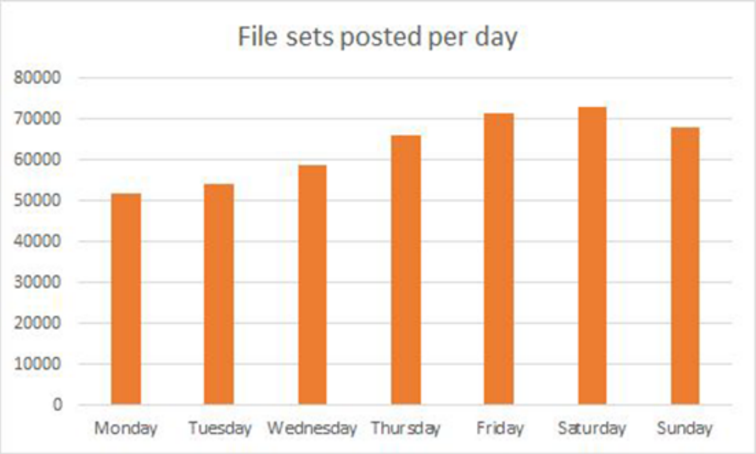

Figure 5: File set posted per day

For the creation of the file sets posted per hour we have taken the file set database and calculated an average of 3,211,532 records. We have chosen to display this data in percentages instead of an article count because it relies to much on the day in question. We cannot see a clear distinction between night and day. This is likely due to large scale automation and the world wide userbase.

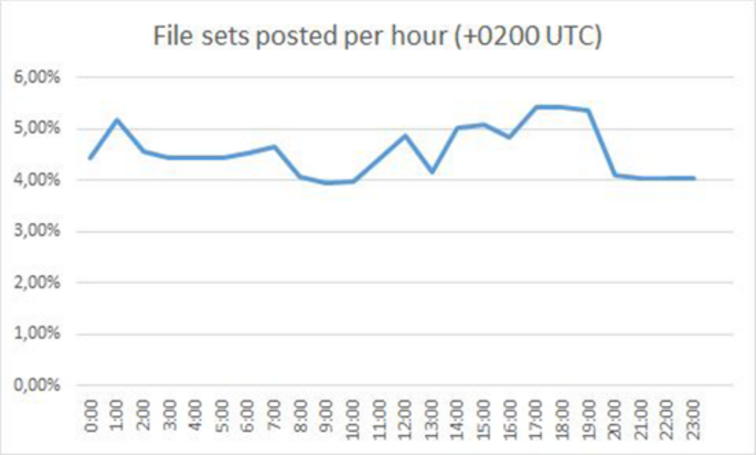

Figure 6: File set posted per hour

4.5.5 Removal time

We have observed the time between an article being posted and the time of the first unavailable record from the EFnet subset. The EFnet subset was chosen because otherwise the scan interval would be to great. The EFnet subset was checked every 5minutes for the first day and every hour for files older then a day. We have only been able to-do this between the time we started and stopped checking. As no reliable data can be retrieved without having the fileset available. We have plotted the time differences in containers of one hour and created a frequency graph. The source data is available in the download section.

For example the provider: USENET server. We can see that for the first 18hours of an fileset being posted to USENET no articles are removed. Only after this time we can see a spike of filesets being removed. And we can see a small spike after 1 day after which very little is removed.

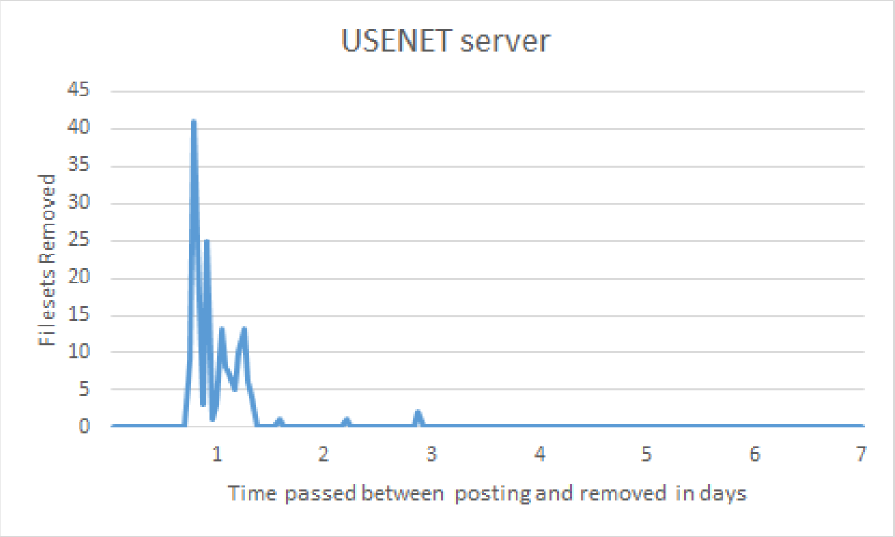

Table 12: Removal statistics seconds between posted and removed

Provider	Avarage	Median	Mode	Range
Astraweb	342878	437063	601643	689
Bulknews	119336	105360	127243	150
Eweka	89902	77516	87977	136
Fast Usenet	246325	206028	561145	718
Giganews	310761	307659	542508	143
Hitnews	84732	77016	106875	169
NEXTGEN	0	0	0	0
Sunny usenet	86682	77218	108448	170
USENET server	84932	76977	106407	169
UseNeXT	186260	99412	495955	689

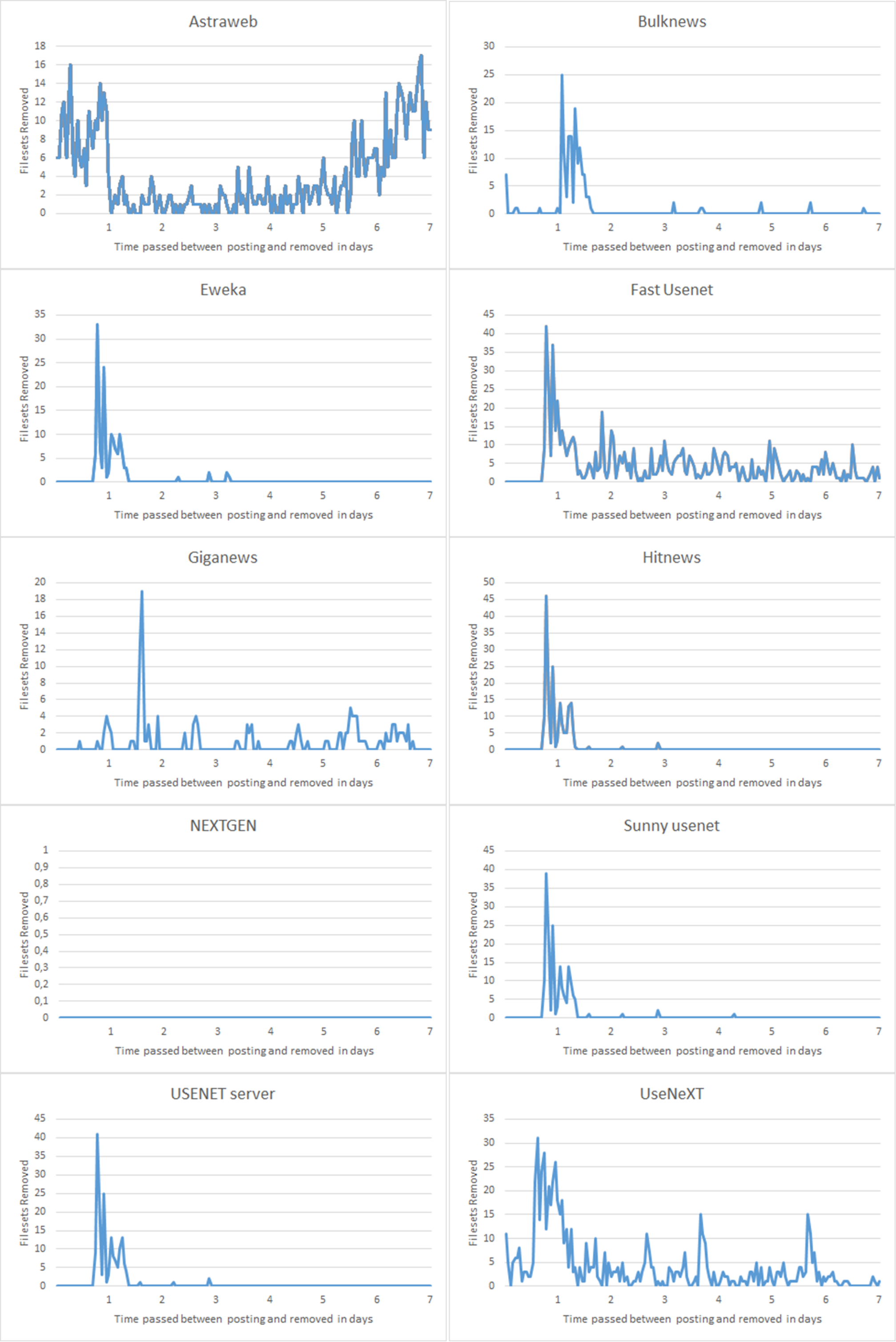

Figure 7: Removal behavior per provider

4.5.6 Search Engine

A search engine has been constructed to be able to search through the collected data. The proof of concept web page is located at `http://nzb.ninja/` The site offers the ability to search the collected subjects. At time of writing the site is only available from inside the SNE subnet.

4.5.7 Correlation between providers

As shown in the graphs above there are very similar removal patterns between several USENET providers. Several have the same parent company according to Internet sources as shown in Table 3 on page 10. It is therefore likely that they are using the same server infrastructure and likely share DCMA takedowns. We have calculated the correlation between (A) the availability of each article in the two datasets, and (B) the time an article was available before it was removed .

Table 13: Correlation between providers solely on availability of articles

provider	Astraweb	Bulknews	Eweka	Fast Usenet	Giganews	Hitnews	NEXTGEN	Sunny usenet	USENET server	UseNeXT
Astraweb	1.00									
Bulknews	0.15	1.00								
Eweka	0.15	**0.88**	1.00							
Fast Usenet	0.17	0.14	0.17	1.00						
Giganews	-0.02	0.05	0.06	-0.01	1.00					
Hitnews	0.21	**0.76**	**0.84**	0.20	0.07	1.00				
NEXTGEN	0.00	0.00	0.00	0.00	0.00	0.00	1.00			
Sunny usenet	0.19	**0.69**	**0.77**	0.20	0.06	**0.91**	0.00	1.00		
USENET server	0.21	**0.76**	**0.84**	0.20	0.07	**1.00**	0.00	0.91	1.00	
UseNeXT	0.14	0.22	0.22	0.21	0.00	0.27	0.00	0.25	0.27	1.00

Table 14: Correlation between providers by time to unavailable. 50% and higher in bold

Provider	Astraweb	Bulknews	Eweka	Fast Usenet	Giganews	Hitnews	NEXTGEN	Sunny usenet	USENET server	UseNeXT
Astraweb	1.00									
Bulknews	-0.62	1.00								
Eweka	-0.09	0.22	1.00							
Fast Usenet	-0.19	0.05	0.50	1.00						
Giganews	-0.21	-0.02	0.12	0.06	1.00					
Hitnews	-0.10	0.18	**0.86**	**0.58**	0.13	1.00				
NEXTGEN	-0.03	-0.01	0.00	0.00	0.00	0.00	1.00			
Sunny usenet	-0.11	0.18	**0.85**	**0.58**	0.13	**1.00**	0.00	1.00		
USENET server	-0.10	0.18	**0.86**	**0.58**	0.13	**1.00**	0.00	**1.00**	1.00	
UseNeXT	-0.67	0.25	0.20	0.16	-0.04	0.19	-0.01	0.19	0.19	1.00

4.5.8 Trends in file set names

We have done an analysis of which words are most common in the subject field of the file collections. The entire table is available to download in the download section.

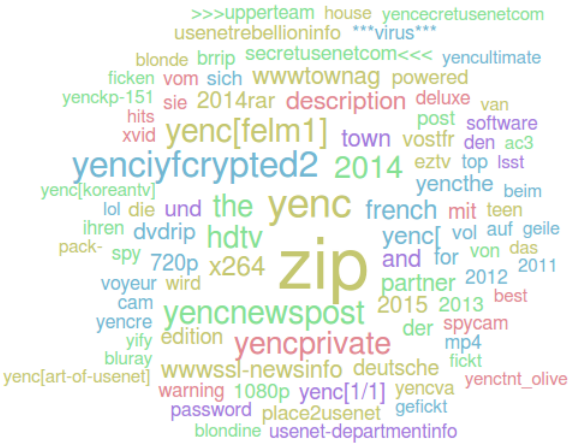

Figure 8: Word Cloud of file set names

We have created a Word Cloud for the subjects that have been removed from the USENET.

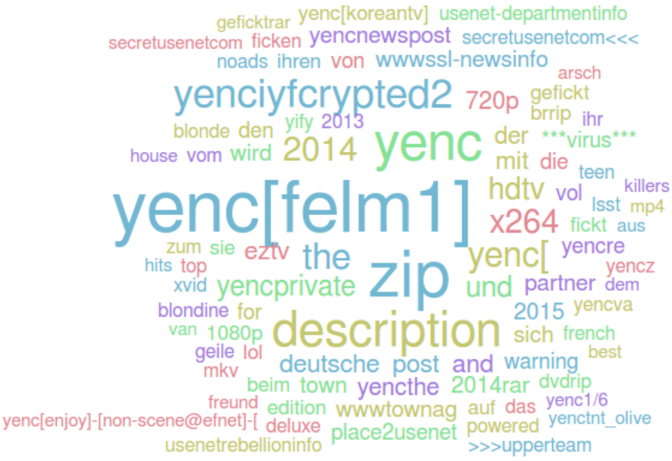

Figure 9: Word Cloud of not found file set names

We see several encoding/encryption methods prominently displayed in the wordcloud: yenc[felm1], yenciycrypted2 and yencprivate. This is a method used by pirates to hide from copyright authorities. However it seems that the content does still gets removed.

5 Conclusion

During the project we have created a system that is able to scan part of USENET for articles being posted. This means that we have a comprehensive database of what is being put on USENET. By making use of the lightweight STAT command we are able to quickly check which articles are still available and which have been removed. The speed at which we can do this highly depends on the provider and subscription with this provider.

The feasibility of checking the entire USENET for removed articles depends on how far you are willing to go back in time to check articles and it depends on your definition of up-to-date. It is, however, feasible to do this kind of checking; we are able to check one year of posted articles in less than 56 hours. With a larger budget for USENET accounts we would be able to do more concurrent checks increasing the frequency of checks.

The data that has been gathered during the experiment has yielded some interesting results. There is a substantial difference in article availability between providers. The data shows that removal time differs per provider. We also see that some resellers have exactly the same article availability as the source, however, there are exceptions to this rule. It is likely that some resellers have caching servers that require them to pay less to access the source server for frequently accessed articles.

The project has shown that it is possible to create a search index that is aware of availability per provider. The limited number of providers with their own storage makes it possible to create a search engine that is aware of all article availability of all the USENET providers.

6 Future Work

6.1 Compare copyright holders

It would be interesting to investigate the different copyright holders and see if there is a difference between the copyright holders. And if all works receive the same attention.

6.2 Predicting takedowns

It would perhaps be possible to teach a machine with this data set to predict which filesets are copyrighted material. And indicate which removal could possibly be illegitimate.

6.3 Larger retention & run for longer

Currently we have a limited retention in our index and availability records. We would like to grow our database to match the retention of the providers. This would give a complete view of the USENET and which articles are available. Give an accurate overview of the state of USENET.

References

[1] Astraweb. *I see copyrighted material on usenet. What should I do?* URL: http://helpdesk.astraweb.com/index.php?_m=knowledgebase&_a=viewarticle&kbarticleid=7.

[2] Giganews. *The Digital Millennium Copyright Act (DMCA).* URL: http://www.giganews.com/legal/dmca.html.

[3] harley.com. *HARLEY HAHN'S USENET CENTER File Sharing Tutorial.* URL: http://www.harley.com/usenet/file-sharing/06-the-limitations-of-usenet-file-sharing.html.

[4] Jan Lindstrom MariaDB. *Performance evaluation of MariaDB 10.1 and MySQL 5.7.4-labs-tplc.* 2014. URL: https://blog.mariadb.org/performance-evaluation-of-mariadb-10-1-and-mysql-5-7-4-labs-tplc/.

[5] newsgroupservers.net. *Usenet Companies and Mega Server Cross Reference.* 2012. URL: http://www.newsgroupservers.net/newsgroup_server_resellers.

[6] nzbusenet.com. *Fixing Usenet DMCA problems.* URL: http://www.nzbusenet.com/fixing-usenet-dmca-problems/?lang=en.

[7] nZEDb. *nZEDb Ubuntu Installation Web Guide.* 2015. URL: https://github.com/nZEDb/nZEDb_Misc/blob/master/Guides/Installation/Ubuntu/Guide.md.

[8] PhoneR@nger.HiHo. *yEnc tools.* URL: http://usenethelp.codeccorner.com/yEnc.html.

[9] usenet providers.net. *Back-end Usenet Provider and Usenet Resellers.* 2014. URL: http://www.usenet-providers.net/newsgroup-resellers.php.

[10] Usenext. *Digital Millennium Copyright Act Notification Procedure.* URL: http://www.usenext.nl/dmca/.

[11] vergelijkusenetproviders.nl. *Usenet Provider Lijst.* URL: https://vergelijkusenetproviders.nl/usenet-provider-lijst/.

7 Appendix

A Download section

Homemade software

`http://nzb.ninja/scanners/`

Unavailable file sets in csv format

All subjects that are unavailable on atleast one provider [35MB]
`http://nzb.ninja/data/Unavailable.CSV`
The EFNET subset that are unavailable on atleast one provider [5MB]
`http://nzb.ninja/data/Unavailable_efnet.txt`

Word frequency table data

Word count data unavailable records [7MB]
`http://nzb.ninja/data/wordcloud_unavail.txt`

Performance report

Python time spent on line report [11KB]
`http://nzb.ninja/data/performance_report.txt`

Full database

A complete backup of the database [60956MB]
`http://nzb.ninja/data/database_bck.tar.gz`

B USENET Map

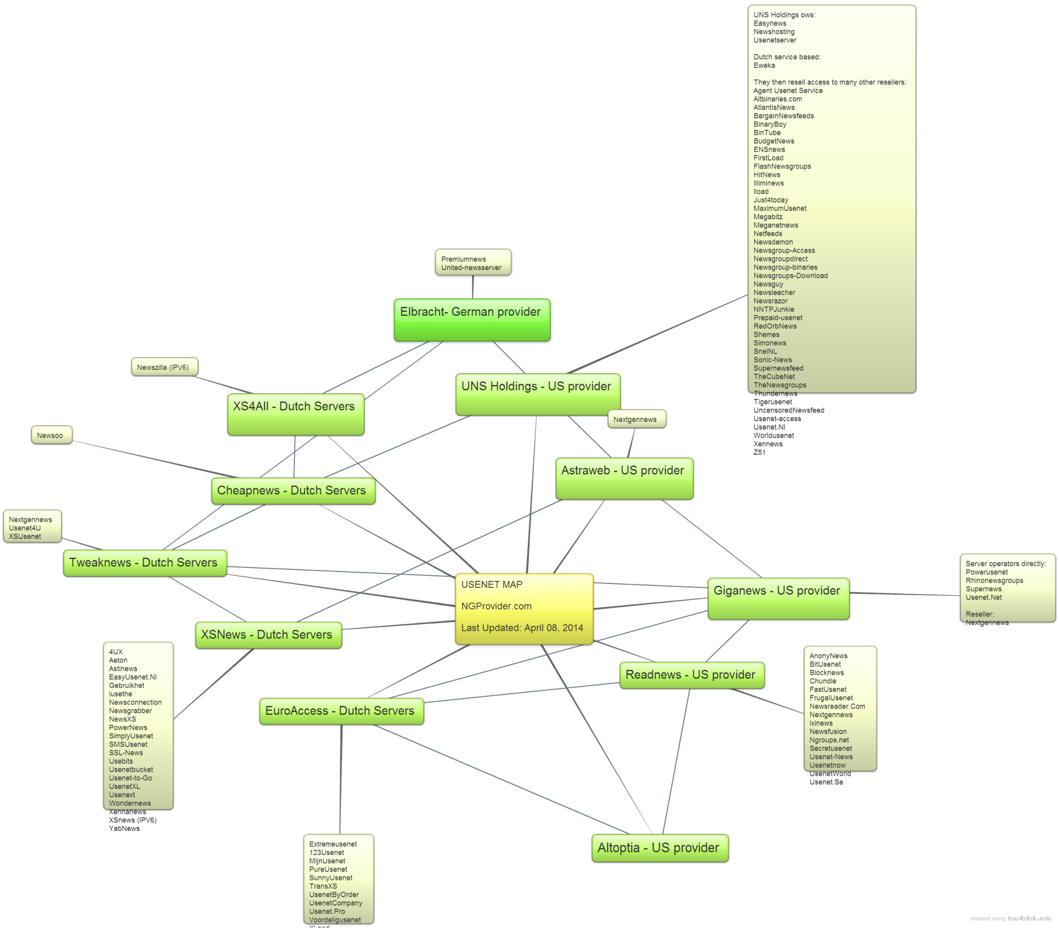

Figure 10: USENET Providers and thier connections [9]

C USENET providers cross referenced [11] [5] [6]

Usenetprovider	Reseller
Astraweb	Astraweb
Bintube	Astraweb
UseNeXT	Aviteo Ltd
Cheapnews	Cheapnews
sonic-news.com	FDC Servers.net
teranews	FDC Servers.net
hitnews.eu	Hosted at RSP Infrastructure, LLC
newsrazor.net	Hosted via Integra Telecom
newsville.com	Hosted via Virtual Interactive Ctr
uncensorednewsfeed.com	Reflected Network
astraweb.com	Searchtech Ltd
GoGoUsenet.com	thumbnails
diiva.com	thumbnails
newscat.com	thumbnails
thumbnailednewsgroups.com	thumbnails
ucache	thumbnails
usenetbinaries.com	thumbnails
xusenet	thumbnails
123Usenet	UNS Holdings
Aeton	UNS Holdings
Agent	UNS Holdings
airnews.net	UNS Holdings
Alibis	UNS Holdings
Alt Binaries	UNS Holdings
Anarqy	UNS Holdings
Anonynews	UNS Holdings
Atlantis News	UNS Holdings
BargainNewsfeeds	UNS Holdings
BinaryBoy	UNS Holdings
BinTube	UNS Holdings
Block News	UNS Holdings
bubbanews.com US	UNS Holdings
CheapNews	UNS Holdings
EasyNews	UNS Holdings
ENSnews	UNS Holdings
Eurofeeds	UNS Holdings
Eweka	UNS Holdings
Extremeusenet	UNS Holdings
fastusenet.org	UNS Holdings
Firstload	UNS Holdings
Flash Newsgroups	UNS Holdings
FlashNewsgroups	UNS Holdings
forteinc.com	UNS Holdings
FrugalUsenet	UNS Holdings
Giganews	UNS Holdings
Hitnews	UNS Holdings
Illiminews	UNS Holdings
iLoad-Usenet	UNS Holdings

Usenetprovider	Reseller
iusenet.com SSL US	UNS Holdings
Ixinews	UNS Holdings
Just4Today	UNS Holdings
Maximum Usenet	UNS Holdings
Megabitz	UNS Holdings
Meganetnews	UNS Holdings
Mijn Usenet	UNS Holdings
Netfeeds	UNS Holdings
News Demon	UNS Holdings
News Fusion	UNS Holdings
News Guy	UNS Holdings
News Hosting	UNS Holdings
News Razor	UNS Holdings
News Reader	UNS Holdings
Newsgroup Direct	UNS Holdings
Newsgroup-Access	UNS Holdings
Newsgroup-Binaries	UNS Holdings
Newsgroup-Download	UNS Holdings
Newsgroupdirect	UNS Holdings
Newsgroups-Download	UNS Holdings
Newsgroups.com	UNS Holdings
newsguy.com US	UNS Holdings
newshosting.comUS	UNS Holdings
NewsLeecher	UNS Holdings
Newsrazor	UNS Holdings
Newsreader	UNS Holdings
NewsServers	UNS Holdings
NGroups	UNS Holdings
NNTP Junkie	UNS Holdings
Pay Less Usenet	UNS Holdings
Power Usenet	UNS Holdings
Prepaid-Usenet.de	UNS Holdings
Pure Usenet	UNS Holdings
Readnews	UNS Holdings
Red Orb News	UNS Holdings
RedOrbNews	UNS Holdings
Rhino Newsgroups	UNS Holdings
Secretusenet	UNS Holdings
Shemes	UNS Holdings
Simonews	UNS Holdings
SnelNL	UNS Holdings
Sonic-News	UNS Holdings
speakeasy.net (ISP)	UNS Holdings
Stealthnews	UNS Holdings
SunnyUsenet	UNS Holdings
Supernews	UNS Holdings
Supernewsfeed	UNS Holdings
Tera News	UNS Holdings
The Cube Net	UNS Holdings
The Newsgroups	UNS Holdings
TheCubeNet	UNS Holdings
TheUsenet	UNS Holdings

Usenetprovider	Reseller
Thundernews	UNS Holdings
Tiger Usenet	UNS Holdings
Titan News	UNS Holdings
TransXS	UNS Holdings
Tweak.nl	UNS Holdings
Tweaknews	UNS Holdings
TweakNews.Nl	UNS Holdings
Uncensored Newsfeed	UNS Holdings
Unison	UNS Holdings
Usenet Binaries	UNS Holdings
Usenet Central	UNS Holdings
Usenet Company	UNS Holdings
XSNews	UNS Holdings
Usenet Rocket	UNS Holdings
Usenet Server	UNS Holdings
Usenet-Access	UNS Holdings
Usenet-News	UNS Holdings
usenet-news.net EU	UNS Holdings
Usenet.net	UNS Holdings
Usenet.NET	UNS Holdings
Usenet.net	UNS Holdings
Usenet.nl	UNS Holdings
Usenet.Nl	UNS Holdings
Usenet.Pro	UNS Holdings
Usenet.pro	UNS Holdings
Usenet.se	UNS Holdings
usenet.se	UNS Holdings
Usenet.se	UNS Holdings
Usenet4U	UNS Holdings
UsenetByOrder	UNS Holdings
usenetcentral.com	UNS Holdings
usenetguide.com US	UNS Holdings
usenetmonster.com	UNS Holdings
Usenetnow	UNS Holdings
usenetrocket.com	UNS Holdings
usenetserver.com	UNS Holdings
Voordelig Usenet	UNS Holdings
WorldUsenet	UNS Holdings
Xen News	UNS Holdings
XLned	UNS Holdings
XLUsenet	UNS Holdings
XSUsenet	UNS Holdings
YottaNews	UNS Holdings
Z51	UNS Holdings
Budget News	XENTECH
HitNews	XENTECH
Xennews	XENTECH

Usenetprovider	Reseller
4UX	XSNews
AstiNews	XSNews
Astinews	XSNews
binverse.com	XSNews
Bulknews	XSNews
Easynews	XSNews
EasyUsenet	XSNews
eurofeeds.com	XSNews
Gebruikhet	XSNews
Newsconnection	XSNews
NewsGrabber	XSNews
NewsXS	XSNews
PowerNews	XSNews
SimplyUsenet	XSNews
SMSUsenet	XSNews
SnelNL	XSNews
SSL-News	XSNews
Surfino	XSNews
Usebits	XSNews
Usenet-to-Go	XSNews
Usenet2Go	XSNews
Usenet4U	XSNews
UsenetBucket	XSNews
UsenetXL	XSNews
Wondernews	XSNews
XSNews	XSNews
YabNews	XSNews