On the privacy of file sharing services

Nick Nikiforakis
Francesco Gadaleta
Yves Younan
Wouter Joosen

Report CW 585, May 2010

Katholieke Universiteit Leuven
Department of Computer Science
Celestijnenlaan 200A – B-3001 Heverlee (Belgium)

On the privacy of file sharing services

Nick Nikiforakis
Francesco Gadaleta
Yves Younan
Wouter Joosen

Report CW 585, May 2010

Department of Computer Science, K.U.Leuven

Abstract

File sharing services are used daily by tens of thousands of people as a way of sharing files. Almost all such services, use a security-through-obscurity method of hiding the files of one user from others. For each uploaded file, the user is given a secret URL which supposedly cannot be guessed. The user can then share his uploaded file by sharing this URL with other users of his choice. Unfortunately though, a number of file sharing services are incorrectly implemented allowing an attacker to guess valid URLs of millions of files and thus allowing him to enumerate their file database and access all of the uploaded files. In this paper, we study some of these services and we record their incorrect implementations. We design automatic enumerators for two such services and a privacy-classifying module which characterizes an uploaded file as "private" or "public". Using this technique we gain access to thousands of private files ranging from private and company documents to personal photographs. We present a taxonomy of the private files found and ways that the users and services can protect themselves against such attacks.

Keywords : privacy, Web 2.0, web services, cloud, file storage, file sharing

On the privacy of file sharing services

Nick Nikiforakis, Francesco Gadaleta, Yves Younan, and Wouter Joosen

DistriNet, Dept. of Computer Science
Katholieke Universiteit Leuven
Celestijnenlaan 200A B3001
Leuven, Belgium
{nick.nikiforakis,francesco.gadaleta,yves.younan,wouter.joosen}@cs.kuleuven.be

Abstract. File sharing services are used daily by tens of thousands of people as a way of sharing files. Almost all such services, use a security-through-obscurity method of hiding the files of one user from others. For each uploaded file, the user is given a secret URL which supposedly cannot be guessed. The user can then share his uploaded file by sharing this URL with other users of his choice. Unfortunately though, a number of file sharing services are incorrectly implemented allowing an attacker to guess valid URLs of millions of files and thus allowing him to enumerate their file database and access all of the uploaded files. In this paper, we study some of these services and we record their incorrect implementations. We design automatic enumerators for two such services and a privacy-classifying module which characterizes an uploaded file as "private" or "public". Using this technique we gain access to thousands of private files ranging from private and company documents to personal photographs. We present a taxonomy of the private files found and ways that the users and services can protect themselves against such attacks.

1 Introduction

In 2006 the first file sharing service, Rapidshare, was founded. Today Rapidshare provides more than 5 petabytes of storage on their servers and can handle up to three millions of users accessing their service simultaneously [11]. Rapidshare spawned a new class of Internet companies called file sharing services. These services of which Rapidshare is the largest best known, are so successful because they fill a basic need of Internet users [1]. This need is the need of sharing data with other users. While P2P networks existed long before file sharing services (e.g. Napster, KaZaa),these networks allowed only a binary type of sharing: a user's files where either shared with everyone on the network, or with no-one. Emailing files as a means of sharing was and still is a choice, however even the most generous email providers do not allow attachments that are larger than 10-20 Mbytes. Emailing attachments is generally also quite slow to both send

[1] While we use RapidShare to illustrate what a File Sharing Service is, this sharing server is not included in our privacy enumeration study

and receive for users and causes significant strain on all the mail servers that the files must pass through.

On the other hand, file sharing services allow uploads to their service of much larger files (e.g. Rapidshare allows files to be 200MB in size). Once the files are uploaded, it provides the user with a unique URL for each uploaded file which he can later use to access them. If the files are meant to be publicly accessed, these URLs are posted to websites, forums and IRC channels. On the other hand, if the files are meant to be private and shared only with a limited amount of users, the links are usually shared through email or Instant Messaging platforms.

Each such link is private and file sharing services state that their service can't be "searched" meaning that unless someone has the exact link to an uploaded file, he can't access that file. For every file that a user uploads, the service creates a token included in the final URL, which is given to the user.

Access to the file thus usually depends on knowing the correct URL to access the file (i.e. it depends on knowing the token). Thus access control is performed by providing the URL or token to another user or not. However, many of file sharing services provide tokens which are not at all random, allowing an attacker to guess the URL and access the files which they are not meant to access.

In this paper, we are going to explore the ramifications of such design decisions and how they jeopardize the privacy of users. We will present two case studies of websites hosting millions of files and how we got access to them. For every file that we enumerate, we use Yahoo search as a privacy engine. If the filename returns search results, then the file was meant to be public. If not, then the file is most likely private and probably contains sensitive content. We present a taxonomy of all the files recorded on the hosting providers, we found that up to 1/5th of them can be considered private files. Of those private files, we manually examine the contents of several files that could be interesting based on file name and extension. This allowed us to find many different files which are considered sensitive by either individuals or companies: medical records, salary lists, lists of people who applied for jobs including the scores they received in their interviews, budgets for companies, etc.

2 File sharing and cloud services

The driving force behind our work, is to demonstrate that what users understand as private and secure can be interpreted very differently by file sharing and cloud services. A modern trend in computers, is moving everything to the cloud. Hundreds of companies have emerged in the last couple of years providing cloud services and some go as far as proposing that every application and all files should be stored in the cloud, e.g. Google ChromeOS [3]. This mentality can and already is risking the privacy of user data. A user must be able to understand exactly how the cloud works and if a company does what it claims in order to be able to make an educated decision of whether his files must move to the cloud and if yes, to where and in what format. While users generally have an understanding of the fact that files stored on their computers are - in general - only accessible by people with physical access to the computer, it is a lot harder to figure out who has access to which files, because this can differ significantly depending on the cloud service that the user uses and users do not always read the EULA, but rely on their experience with one service to carry over to the next.

3 Life cycle of files on File Sharing services

In this section we will describe the typical life cycle of a file in relation to file sharing services and point out the privacy-sensitive steps. In order to model this life cycle we assume a file sharing service (FSS) which allows users to upload files, without the need of accounts, that provides the user with a unique URL which can be later used to access the uploaded file and that a file is deleted from the service when a) the file is inactive for n days or b) the file was reported as a copyright violation.

1. Content is created. Content can be created either by dedicated multimedia devices (such as digital camera or video camera) or from software that runs on a user's computer.
2. The user who created the content wants to share it with other users. In the context of this paper, the user decides to share this newly-created content (from here on called a file) using a file sharing service.
3. He chooses a file sharing service and requests the upload of that file.
4. One of the service's webservers, receives the file and stores it in its cloud. It creates a unique token and binds it to the uploaded file
5. The unique token is returned to the user and he is instructed to use it every time that he wishes to access his uploaded file.
6. The user now distributes this token according to the nature of his file. a) If the file is meant to be public, the user posts the link on publicly-accessible webpages such as forums, newsletters and social networks. b) If the file is meant to be private then the user shares the token for the file using end-to-end communication tools such as emails or instant messaging platforms.
7. The public or private recipients of the token, use it to access the file that the user originally uploaded.
8. If the file violates copyright laws (movies, music, non-free software) then a third user can possibly report it. In this case the file is deleted and the unique token is possibly reused.
9. If the file survives this process it will be accessible until a certain in-activity threshold is reached in which case it is removed by the FSS in order to save space in its cloud.

In the context of this paper, we are mainly interested in steps 4 and 6b. The user uploads a private file to a FSS and uses an end-to-end way of communicating the unique copy returned by the service to the intended recipients of the file. We assume that the trusted recipients of the file token will not forward it to other untrusted users. In this scenario, the only way that the file can be accessed by an untrusted user is if the user manages to guess the unique token returned by the FSS. Unfortunately in many cases, this is as easy as subtracting "one" from a valid unique token.

4 Enumerating File Sharing Services

In order to understand how different file sharing services manage their files we compiled a list of the top such services and we uploaded multiple files to each such service, searching for patterns in the resulting URLs. In the following table, we list eight well-known FSS (marked as FSS1 to FSS8 in order to protect their service from misuse) and we present two real tokens that were assigned to two consecutive uploads of the same file:

Service	**1st Token**	**2nd Token**
FSS1	358831587/llncs2e.zip.html	358831730/llncs2e.zip.html
FSS2	b0790a8/n/llncs2e.zip	b0790ae/n/llncs2e.zip
FSS3	15738111/llncs2e.zip	15738117/llncs2e.zip
FSS4	1909452917/llncs2e.zip	1909452920/llncs2e.zip
FSS5	1878483	1878484
FSS6	34249	34250
FSS7	gxmif2	33kpv6
FSS8	jzjztjkhjty	mz5lo0hwm4l

The first six FSS appear to return a URL which consists of an incremental token and, in the case of the first four, the name of the uploaded file. By testing the URLs without the filenames we found out that only FSS1 enforced the use of it, meaning that in FSS2 to FSS6, a user can request a file without providing the filename. This enables an attacker to enumerate through their entire file database without having to guess possible filenames. On the other hand, FSS7 and FSS8 generates tokens that appear to be random. In these filesharing websites, if the token truly is random, an attacker can only guess valid URLs through the use of brute-forcing. However, in both cases, the fact that the filename is not required, allows a user to iterate over all possible uploads by using the token. FSS8 uses 11 characters for the token, making the search space 130 quadrillion possible files, which makes it impractical to enumerate over the files. However, FSS7 only uses 6 characters, making the search space only 2 billion large. This allows an attacker to retrieve all files from FSS7 by enumerating over these 2 billion combinations, making an attack on this service more practical.

From the aforementioned FSS we decided to enumerate FSS2 and FSS4. Both of them generate incremental tokens for each uploaded file and their tokens reveal that users have uploaded many millions of files on their servers. For FSS4, enumeration consists of starting from a valid token integer and continuously subtracting "one" to get to the previous valid token. FSS2 is also incremental but instead of using integers, they use an alpha-numerical string were each digit can be a number (0-9) or a letter (a-h). We created two enumerators, one for each service and configured them to request for URLs using a new generated token each time. We limit our requests to a rate of one per 5-10 seconds , in order to avoid overloading the file sharing services. Given the popularity of both FSS, our requests are interleaved with tens of "legitimate" requests, which enabled us to enumerate their services for more than two weeks without any problems.

These enumerators record valid URLs along with the filename and the file size (as reported by each FSS). While these lists are sensitive by themselves there is little interest in files that are publicly available since the uploaders intended those files to be shared. In this paper, we are mainly interested in examining the privacy risks of FSS and thus we need to differentiate between files which were intended to be public and files intended to be private.

We decided to use search engines as our classification mechanism. Their ubiquitous nature makes them a good mechanism of classifying data as public. If the filename of an enumerated file returns search results, then the user who uploaded that file probably posted it on a publicly accessible resource (e.g. blogs, forums, websites etc.) On the other hand, if the filename doesn't return search results, then the file was exchanged in a more private manner (e.g. private forums, Instant Messaging, Email). These files were not intended to be public and they could possibly contain sensitive data.

5 Evaluation

In this section we will evaluate our enumeration of two file sharing services (FSS2 and FSS4) by measuring several attributes of all uploaded files and then focus on the ones reported as private.

5.1 Public & Private File Taxonomy

Using our two enumerators from three different hosts for two weeks, we gained access to a total of 313,000 files. In Figs. 1,2 we present the percentages of the top ten file extensions from the files that FSS2 and FSS4 host on their servers. It is interesting to see that in both services, `.rar` extensions are the most prevalent with a significant difference. `RAR` files are compressed archives that have gained increasing popularity since they allow the user to easily split up a large file into smaller counterparts, with the possibility of creating password-protected archives. This observation is in accord with the observation that file-sharing websites are primarily used for illegally distributing copyrighted material [12, 13, 8]. Copyrighted material, such as movies and games, are usually split up into parts in order to make it past the FSS's maximum file size. Thus, a standard sized movie of 700MB will produce a number of `rar` files which will all be uploaded to a FSS, explaining the large percentage of such files in our measurements.

Fig. 3 shows the combined percentage of each file hosted on those two FSS, according to the file sizes. Each point on the x-axis is a "greater-or-equal" bucket, e.g the first point in the x-axis signifies the files that are greater or equal to 200Mbytes. While the values of the two services appear to be quite different, an interesting observation is that they are much more similar if we shift the

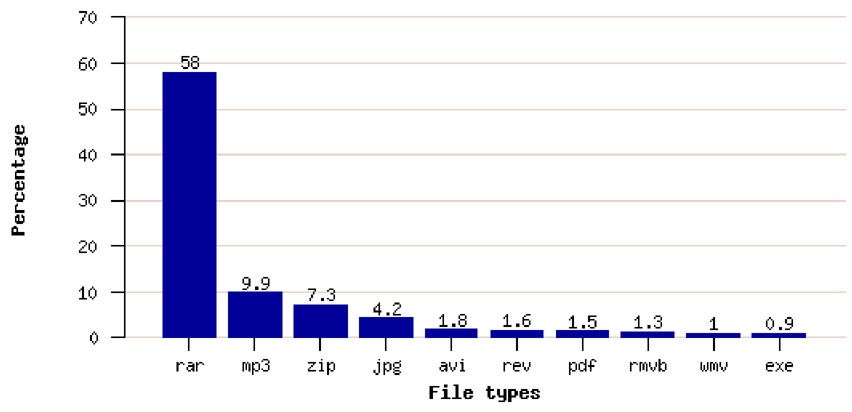

Fig. 1. Taxonomy of 83,000 unique files, enumerated from FSS2

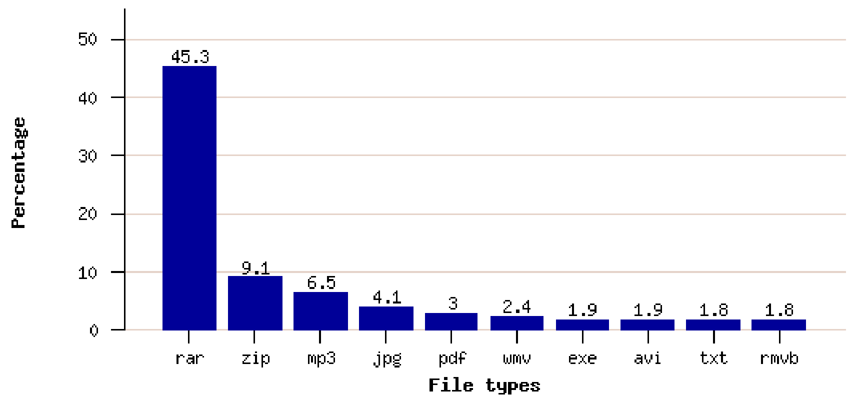

Fig. 2. Taxonomy of 230,000 unique files, enumerated from FSS4

FSS2 values one step to the right. This effect is because, FSS4 has a maximum-upload size limit of 200Mbytes while FSS2's limit is 300Mbytes enabling users to upload files that are larger than 200Mbytes. Despite that however, the sizes of files stored on the two FSS are quite different especially when the filesize decreases, revealing that apart from distributing large multimedia files, the two services are being used by two distinct sets of people for distinct purposes.

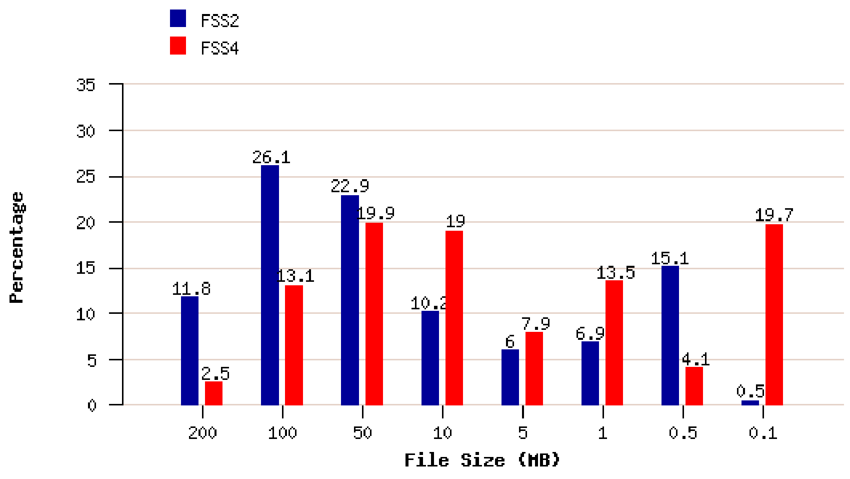

Fig. 3. Taxonomy of the file sizes from both FSS

5.2 Private File Taxonomy

From the total of 313,000 files that we enumerated, we removed all files that had one of the following extensions: `mp3,exe,wmv,3gp,rar`, resulting in 128,000 files. Most of these files extensions belong to uploaded files that are pirated music, software or movies and since we are mostly interested in privacy-critical data we decided not to classify such files. Each resulting filename is going to used as a search parameter at Yahoo search. Unfortunately the rate at which we can access the search API of Yahoo is much lower than the rate at which we enumerated the two FSS but even so, at the time of this writing, our privacy-classifying module has reported 22,000 files as private, meaning that Yahoo returned zero results when we searched for them.

In Fig. 4 we present the taxonomy of the file types of these 22,000 private files. Interestingly, two of the file types that we consider sensitive, `jpg` and `pdf`, are in the top 5 places of the graph.

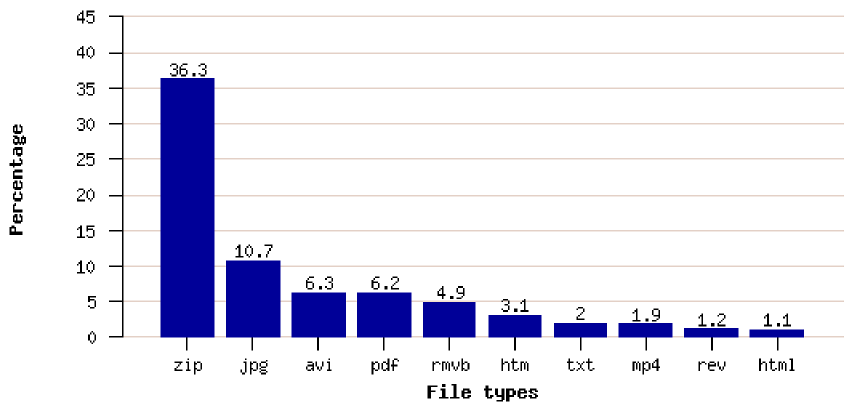

Fig. 4. Taxonomy of file types from a population of 22,000 private files

5.3 Inspection of Private Files

In the interest of understanding how many of the files reported as private where actually private (in the sense that they hold sensitive data) we downloaded the first 100 private files from 6 categories of potentially sensitive types of files and inspected them manually. In the following table we list the results from our manual inspection:

File type	**Description**	**Sensitive Content**
jpg	Image data	94%
xls/xlsx	Spreadsheet data	58%
doc/docx	Documents	25%
png	Image data	4%
txt	Text	2%
pdf	Portable Document Format	0%

Photographs shot by digital cameras have usually a `.jpg` file extension and our manual inspection confirms this. Almost all of the download `jpg` files were pictures of people and places that were obviously meant to be privately (if at all) shared. Moving down the file type list we found spreadsheets and documents that contained private data, such as bank transfers, email addresses and phone numbers. Most `png` files were commercial images and most text documents contained links for new files hosted on FSS. Even though we were expecting that many files with the `pdf` extension would be private we were surprised when we found out that all of them were either e-books or academic papers. In the following list, we describe some of the most sensitive files that we found:

– Service manual of a digital photo printer manufacturer containing the administrator password

- Bank Statements for several banks
- Budget for 2009 and salary break-up for a specific company
- List of employees of a company, along with their addresses, emails and mobile phone numbers
- A copy of a death certificate
- Several recommendation letters
- Documents containing transcribed audio of consultations of a doctor

All of the above information can be used by attackers for social engineering, information gathering or even identity theft.

6 Discussion

It is evident that file sharing services can be the cause of a serious private-data leakage. We proved that an attacker with the most basic programming skills can get access to millions of online files in a matter of weeks. File-sharing is a true need and it will always be of value but some extra security measures have to be added to ensure that they can't be misused. While we were searching and testing possible FSS we observed that some, gave the user the option to make the file accessible only through the use of a password. While this measure by itself can be of value, we know that users tend to use simple guessable passwords and/or re-use them across services. We believe that security measures must not rely on end-users in order to be effective and thus we propose a password system that is on by default and that operates solely server-side.

In an imaginary FSS where an uploaded file could be accessed, e.g. through the following URL: `http://www.myfss.com/file/456743/` we propose the addition of an extra field which would be a randomly generated password: `http://www.my fss.com/file/456743/p/mg3957sdaqs`. Using this method, the user is not required to remember any passwords, just copy the link to the file, an action that he already does. If a user (possibly an attacker) requests the file without the use of a password, then the FSS would request that password before giving the user access to that file. Storing an extra identifier for each uploaded file, has of course some storage overhead but we believe that FSS who offer terabytes of online storage wouldn't mind storing a few tens of megabytes of identifiers.

7 Related work

Cloud Computing is a term that resembles the concept of Computer Utility [10] and Software as a Service (SaaS). The advantages of SaaS to both users and providers are well understood [2]. Service providers reduce efforts in software maintenance and versioning because of the centralized control of the software life cycle; users can enjoy the service *anytime, anywhere*, sharing resources and data in a transparent fashion. Despite the amount of definitions bloggers and researchers have given [5, 9, 6, 7], Cloud Computing allows the deployment of Software as a Service without owning or provisioning a datacenter. Thus, a small company with innovative ideas for new Internet services does not need to be concerned about the costs of the required hardware. Moreover, companies with large batch-oriented tasks can make use of the Cloud and its resources to get results quickly, when their programs can scale, since using a high number of servers for a short period would have the same costs of using a more limited hardware for a longer period. Since Cloud Computing refers to both the services and the hardware to provide them, several advantages have been highlighted [2]:

1. the illusion of infinite hardware resources available on demand
2. the elimination of an up-front commitment, which allows companies to start small and increase resources on their needs
3. pay-for-use of computing resources, which allows companies to economize on hardware and software resources after start-up
4. pay-separately per resource, for those services which do not make equal use of computation, storage and network bandwidth

Unfortunately, these benefits might come with a cost in terms of privacy and security. A recent study, accomplished by CSA (Cloud Security Alliance), focuses on security issues amplified by the nature of Cloud Computing and reports the top six threats in order to provide a context to assist organizations in making decisions regarding their Cloud adoption strategies [1]. We report three of the most common threats affecting services in the Cloud.

Abuse of the Cloud The illusion of infinite hardware resources is an attractive feature to spammers and malicious code authors. Several botnets have abused services and resources in the Cloud to spread malware and spam users [16]. A stricter initial registration and monitoring of customer network traffic is considered an effective re-mediation to this insidious threat.

Insecure interfaces Cloud computing providers expose APIs their customers use to interact with Cloud services. The security of the overall system often depends on the security of those APIs. These interfaces are required to be properly designed to provide a secure access to services and resources and protect against malicious users or accidental attempts to circumvent security policies [14].

Data loss or leakage Insufficient authentication or authorization controls may lead to compromise data. The threat of data loss or leakage increases in the Cloud, due to the higher number of interactions in comparison to private datacenters. An interesting solution is proposed in [4] which protects data by encryption and by Wuala [15], a secure online storage service which uses a combination of private datacenter and Cloud Computing together with secure authentication and authorization control policies.

8 Conclusion

In this paper, we studied file sharing services and how they are used for the exchange of public and private files. We observed the way they worked and we identified exploitable weaknesses in the services' generation of random file tokens. We created enumerators for two such vulnerable services, recorded 313,000 files, and found that about 1/5th of these files could be classified as private. While the majority of those files were most likely related to illegal software, many files which can be considered sensitive data were also discovered. These files range from transcribed medical records (transcribed from an audio recording by a medical transcription company), company budget information, salary lists, lists of people who applied for a job, etc. We demonstrated the problem of such services, showed why users must not blindly trust third parties to store their data, especially their sensitive (private) data, and showed that users' expectation of privacy does not carry over from their personal computer to the web as easily. Finally, we proposed a viable solution which ensures that files stored in the cloud of FSS can be accessed only by their rightful owners or users that the owner trusts.

References

1. CSA Cloud Security Alliance. Top Threats to Cloud Computing. `http://cloudsecurityalliance.org/pr20100301a.html`, 2010.
2. Michael Armbrust, Armando Fox, Rean Griffith, Anthony D Joseph, Randy H Katz, Andrew Konwinski, Gunho Lee, David A Patterson, Ariel Rabkin, Ion Stoica, and Matei Zaharia. Above the clouds: A berkeley view of cloud computing. Technical report, 2009.
3. Official Google Blog: Introducing the Google Chrome OS. `http://googleblog.blogspot.com/2009/07/introducing-google-chrome-os.html`.
4. Roxana Geambasu, Tadayoshi Kohno, Amit Levy, and Henry M. Levy. Vanish: Increasing data privacy with self-destructing data. In *Proc. of the 18th USENIX Security Symposium*, 2009.
5. Hamilton J. Perspectives. `http://perspectives.mvdirona.com`.
6. Rangan K. The Cloud Wars 100 billion dollar at stake, 2008.
7. Siegele L. Let it rise: a special report on corporate it. *The Economist*, 2008.
8. Eric Lai. What's replacing P2P, BitTorrent as pirate hangouts? `http://www.thestandard.com/news/2009/10/09/whats-replacing-p2p-bittorrent-pirate-hangouts`.
9. Carr N. Rough Type. `http://www.roughtype.com`.
10. D. PARKHILL. The challenge of the computer utility, 1966.
11. Rapidshare: 1-CLICK Web hosting. `http://rapidshare.com/wiruberuns.html`.
12. Torrent crackdown pushing pirates towards file hosting. `http://www.channelregister.co.uk/2009/10/13/warez_hosting/`.
13. Q&A: Eleven questions for a warez site owner. `http://www.computerworld.com/s/article/9050203/Q_A_Eleven_questions_for_a_warez_site_owner`.
14. websense. P0wning The Programmable Web. `http://securitylabs.websense.com/content/Blogs/3402.aspx`, 2009.
15. Wuala. Wuala Secure Online Storage. `http://www.wuala.com`.
16. ZDNet. Zeus crimeware using Amazon's EC2 as command and control server. `http://blogs.zdnet.com/security/?p=5110`, 2009.