Technology and the Right to be Forgotten

Author:
M. M. Vijfvinkel
Studentnumber:
s4077148

Supervisor:
Dr. J.H. Hoepman
Second Reader:
Dr. G. Alpár

Radboud University

Master's Thesis
Computing Science
Radboud University Nijmegen
July, 2016

Abstract

Personal information is increasingly stored for an indefinite period of time. This changes the norm from forgetting-by-default, as experienced in the human brain, to a norm of remembering-by-default experienced through technology. Remembering by default can have negative consequences for individuals, e.g. personal information can be viewed out of context, or lead to self-censorship. The upcoming General Data Protection Directive seeks to improve on this situation by providing people with a Right to be Forgotten, such that they can request an organisation to erase their personal information. In this thesis we provide insight in how the Right to be Forgotten can be implemented by technology, such that technology adheres more to the norm of forgetting-by-default. We aim to achieve this by analysing the Right to be Forgotten from legal and socio-philosophical perspectives. We use the analyses to investigate what methods of erasure could be used by technology, and which factors (e.g. time) could be used to determine if data should be erased. Furthermore, we provide a classification for technical concepts and explore two technical approaches in depth. Specifically, Freenet which deletes least recently used data, and Volleystore which implements a naturally decaying storage medium.

Keywords: Right to be Forgotten, Right to Erasure, Freenet, Volleystore, Parasitic Data Storage

Contents

1 Introduction

The amount of data currently stored is immense and still increasing[1] [1]. Estimations made in 2002 have shown that five exabytes of data are stored every year with an annual growth rate of 30 percent [2]. The following technological advances have helped create an environment that allows for increasing quantities of data being stored [2]: digitisation, increase in storage capacity, ease of finding information, and access to data. We explain this in more detail below. Additionally we explain several reasons that sustain this environment and the consequences it can have for an individual.

1.1 Background

Digitisation of analogue media to digital media brings several advantages. It makes noiseless copying of data possible, which means that there is less deterioration of data quality during copying. By making use of standardisation, digitisation has also contributed to a more convenient use of different types of data on different devices, and made it possible for data to be transmitted across different networks. For example, photos, videos and text used to be stored on separate media, but these can now be stored on one storage medium, e.g. a hard disk, which can be used in different devices.

Increasing storage capacity and decreasing storage costs results in cheaper storage. One example of this is the increasing number of bits per square inch (areal storage density) on hard disks; also known as Kryder's Law[2]. The use of indexing and searching techniques, make finding and retrieving specific data from a vast amount of storage space easier and more efficient. The Internet makes it possible for people to access stored data around the globe without having to travel to the physical storage location. Also broadband connection speeds are increasing, while prices remain relatively stable, reducing communication costs.

These advances, especially efficient searching and worldwide access, have made it easy for people to share and access data, and information; which in turn encouraged people to share and access data, and information [2].

1.1.1 The value of data

Another reason why more data is stored, is that data has value. Especially personal data, as we explain below.

Analysing data allows for information to be derived from it. The more information can be derived, the more relevant the data, and therefore data becomes valuable. Linking data or information to an individual makes it personal data or personal information[3]. One area in which personal data is highly lucrative, is personalised advertising.

[1] http://www.economist.com/node/15557443, accessed on 13-10-2014
[2] http://www.scientificamerican.com/article/kryders-law/, accessed on 10-10-2014
[3] Note that we will use the terms personal information and personal data interchangeably.

In personalised advertising, a profile of an individual is being created using his or her personal information. The profile can, for example, contain that individual's address, location or telephone number, but it may also contain personal interests or habits. An advertising company can then use this profile to advertise a specific product to that individual, to make the advertisement more relevant for the individual.

In short, personal data has become valuable, because there are companies whose sole business model is dependent on personal data. It is therefore in the interest of those companies to collect and store as much data as possible.

1.1.2 Types of data

Different types of personal data can be used to sustain these business models: volunteered, observed and derived data [3, 4, 5].

Volunteered data is shared knowingly by an individual, e.g. e-mails, photos, tweets, or comments. The individual (i.e. data subject) is usually personally connected with this type of data. It, *'involves a strong sense of ownership and rights to individual control'* [3]. Usually the individual has provided consent for their data to be used, either by using a service (i.e. implicit consent) for basic use levels, or by agreeing with a policy (i.e. explicit consent) for specific use of the data. However, few people read these policies, and when they deny the policy, they receive no service [3]. Thus people are nudged into 'volunteering' data.

It is also possible to gather personal data by observing a transaction between a person, and an organisation. This is called observed data; e.g. cookies, websites visited, time spent on a website, or location data or financial transaction data. The individual might not be fully aware that this data is collected, or what it could be used for. Therefore, with observed data there is a weaker sense of ownership than with volunteered data, but more *'control to the organization which captured it'*[3].

By analysing, combining or mining the volunteered and observed data, extra information can be derived. This is known as derived, or inferred data; e.g. credit scores, or predictions of preferences. Organisations consider derived data a proprietary asset, and therefore it is not known by individuals, how or what personal data (i.e. profile) is gathered.

1.1.3 Loss of power and control

The advances of technology make it possible to share and interconnect data, with consequence an unknown, but large group, of people can access your personal information. Together with not knowing what data is gathered, sharing and interconnectivity of data results in a loss of power and control [2] over the data by the individual. The reason for the loss is that data is not necessarily shared with familiar people or entities alone, but also with unfamiliar ones.

1.1.4 Remembering in the human brain

The human brain behaves differently compared to digital storage mechanisms. It does not remember everything and it also forgets. This contributes to the loss of power and control, because digital storage behaves differently from what people are used to.

A result of the technological advances is that compared to the human brain[4], larger quantities of data and information can be remembered (i.e. stored) at marginal costs. Moreover, data can now be remembered without error, searched and retrieved from locations around the world. However, where humans forget information over time, by not using it regularly, or when information loses its significance; the systems store our data and information continuously, and do not forget by default.

Additionally, it could be argued that there is no longer an incentive to forget (i.e. delete) data, because it is becoming more difficult and costly than keeping data stored. For example, from the point of view of a person, in a large quantity of stored data it can take an excessive amount of time or effort to find, and then remove an old file; compared to just storing an updated version. From the point of view of a business, deleting data from profiles could result in a less accurate marketing campaign, and thus less profit.

1.2 Problem description

Storing increasing amounts of personal data for an indefinite period of time (i.e. no longer forgetting or deleting it), and the interconnectivity of this data, can lead to a loss of power and control by the individual over its data. This loss can have several consequences.

One of the consequences is that it can lead to data being viewed out of context. To illustrate, Alice posts a picture of herself, drunk, at a party on a social network. Unaware that her profile is public, and can be seen by anyone, she assumes that only her friends can see the picture. A year or two later, Alice has forgotten that the picture is still visible online. When applying for a job, Alice's potential employer searches her name with a search engine. Her profile and the picture pops up, resulting in Alice not getting the job.

A similar example is a convicted criminal who has served his sentence, but is unable to reintegrate (i.e. find work or housing), because he is confronted with his criminal past, which was publicised online before.

It can also lead to self-censorship. For example, Alice could also have decided not to put the picture online, and thus self-censor herself. Even though this sounds sensible in this particular case, self-censorship could also be applied to data other than pictures, e.g. political statements, financial or medical records, e-mails, etc. In that case self-censorship limits the freedom of expression, which is a human right.

[4] We refer to [2] for more information on human remembering, forgetting and overcoming the related problems.

Another consequence of storing data for an indefinite period of time, is data manipulation. It can affect the decision making process and memory of people [2]. For example, it can prevent people from getting over a bad experience if they are reminded of that experience due to data that has been stored. A recent example is a father who was accidentally presented with a picture of his daughter in Facebook's Year in Review. The daughter passed away earlier that year[5]. It is also possible for a person to remember something wrong. This can happen when stored data, its existence long forgotten by the person, is secretly altered. When the data is again reviewed by that person after the alteration, he or she might start to doubt her own memory.

To summarise, the indefinite storage period, interconnectivity, value, and the amount of personal data have led to a position to remember-by-default using technology, in contrast with the forgetting-by-default experienced in the human brain. Personal data that is remembered by default may haunt people later in their lives, even when they have no recollection of such data ever having existed.

One way to address these issues, is to provide people with a legal right to have certain data deleted. The Right to be Forgotten is such a right and part of the proposed European General Data Protection Regulation (GDPR)[6]. However, it is cumbersome if people (i.e. data subjects) have to go through the legal system every time they want personal data, or information deleted; see section 2. Also the manner in which the Right to be Forgotten has currently been applied in case law, shows that deleting data is not as straightforward as it seems.

With its introduction, the Right to be Forgotten has been a topic of debate regarding the freedom of expression [6], because information existing in the public domain could be potentially deleted. This debate is difficult, because arguments can be found that support deletion, as well as arguments for keeping the information accessible. For example, a person fulfilling a public role has a right to privacy, and it can be argued that certain information has to be deleted (e.g. naked pictures of a celebrity) from the public domain. However, an article about a politician which abused tax money for personal gain, will gain less support to be removed. In this thesis we will not focus on the discussion if data should be removed from the public domain or not, and what the impact is on the freedom of expression. It would require a careful balancing act between the concerned legislation, which is out of scope for this thesis.

Instead we focus on another concern that makes forgetting, or deleting data less straightforward, and is raised by the question: when can data, or information be considered forgotten or deleted? For example, can data be considered forgotten or deleted if it cannot be found, but still exists somewhere? What if it still exists, but cannot be accessed? In that sense, what concepts or views concerning forgetting or deleting exist? How do they relate to technology? Can they be translated to a framework? These questions are answered by this thesis.

[5] http://meyerweb.com/eric/thoughts/2014/12/24/inadvertent-algorithmic-cruelty/, accessed on 09-07-2015.

[6] In the current version of the GDPR the right is known as the Right to Erasure; see section 2.2

1.3 Research Questions

The goal of this thesis is to provide insight in the Right to be Forgotten from a technical perspective, which can be used to derive a framework. The framework could be used to change the remember-by-default position in storage technology, to a more forgetting-by-default position, such that storing information meets people's natural expectations. To achieve this goal, we want to answer the research question stated below. The main research question is:

> *In what way can the Right to be Forgotten be implemented by technology, such that it fits our societal expectations about forgetting information more naturally?*

To answer this question, we answer the following sub-questions:

- *What are the concepts, regarding erasing or forgetting information, from the legal perspective?*
- *What are the concepts, regarding erasing or forgetting information, from the socio-philosophical perspective?*
- *What framework or model can be derived from these concepts?*
- *What technology or technical concepts exist?*
- *Can the technology or technical concepts be classified using the framework or model?*

1.4 Outline

The outline of this thesis is as follows. Chapter 2 provides a background on the Right to be Forgotten. It explains the relevant parts of the legislative article (i.e. Article 17 of the GDPR), which stakeholders are involved, and the terminology. Using case law we explain the different points of view regarding data deletion from a legal perspective. Chapter 3 approaches the problem from a socio-philosophical angle, and explains what it means to forget data in the environment of information systems, and what factors could decide to erase data. In chapter 4 we argue that erasure of data in an information system can be considered as a failure to access data, similar to forgetting in the human brain. We summarise several technical concepts that fit this model in chapter 5 and classify them. We pick two concepts from the classification and explain them in chapter 6 and 7. In chapter 8 we discuss the feasibility of technical concepts in relation to the general environment of information systems, e.g. business models. We present our conclusions in chapter 9.

2 The Legal Perspective On The Right To Be Forgotten

In this section we briefly describe the history of the Right to be Forgotten (RTBF), and how it is currently defined in the law. We also elaborate on the Right to Erasure, as currently defined in the proposed European General Data Protection Regulation (GDPR). In addition, we use two examples of case law to explain how the concept of forgetting, or deleting has been applied in practice. We focus on European legislation and case law, because legislation of individual member states can vary. The legal analysis presented in this section helps us define the requirements needed to translate the Right to be Forgotten to a technological framework.

2.1 Stakeholders & Terminology

In order to understand the legal analysis later in this chapter, we first explain some terminology. We use the definitions from the GDPR, because the Right to Erasure (Article 17) is described in the same document.

- Personal data: *"'personal data' means any information relating to a data subject;"* in Article 4 (2) [7].
- Data subject: *"'data subject' means an identified natural person or a natural person who can be identified, directly or indirectly, by means reasonably likely to be used by the controller or by any other natural or legal person, in particular by reference to an identification number, location data, online identifier or to one or more factors specific to the physical, physiological, genetic, mental, economic, cultural or social identity of that person;"* in Article 4 (1) [7].
- Processor: *"'processor' means a natural or legal person, public authority, agency or any other body which processes personal data on behalf of the controller;"* in Article 4 (6) [7].
- Controller: *"'controller' means the natural or legal person, public authority, agency or any other body which alone or jointly with others determines the purposes, conditions and means of the processing of personal data; where the purposes, conditions and means of processing are determined by Union law or Member State law, the controller or the specific criteria for his nomination may be designated by Union law or by Member State law;"* in Article 4 (5) [7].
- Public domain: Data or information, that can be viewed by anyone and is publicly accessible. Usually the information can be accessed through the World Wide Web (WWW).

The distinction between the roles of data controller and data processor is difficult, because it often seems like an entity can fulfil both roles at the same time. To clarify this, the Article 29 Working Party[7] published an opinion [8] about

[7]The Working Party provides advice and opinions on matters regarding data protection within the European Union.

the concepts of controller and processor, based on the definitions of the Data Protection Directive.

A data controller is an entity which *'determines the purposes and means of the processing of personal data'*. This includes, for example, why certain data is collected, and how the data is processed. However, a processor exists only, because the data controller either decides to process data within its organisation, or to delegate the processing to a external organisation [8]. In the first case this means that there is a separate legal entity that acts as a processor on behalf of the controller, within the same organisation. In the second case there are actual separate organisations.

A legal distinction between the two is not necessary for the technical part in this thesis, but it is important to take note that the same, or separate business entities (i.e. company) can fulfil the role of data controller or processor, depending on a specific situation.

As such a data subject can also be a data controller or processor. However, when the processing happens *'without any gainful interest in the course of its own exclusively personal or household activity'*, the GDPR does not apply; see Article 2 of the GDPR [7]. This is known as the household exception, and in this situation the Right to be Forgotten is not applicable.

This means, for example, that Alice, a user (data subject) of a social network, cannot legally obtain the Right to be Forgotten from Bob, another user, who uploaded a picture of Alice (e.g. a group photo). (She could of course ask Bob to remove the photo.)

2.2 The Right to be Forgotten

As part of the legal analysis we describe the history, and current interpretation of the RTBF.

2.2.1 History

The RTBF is not a new concept, and is already part (though not explicitly named as such) of current data protection legislation. In fact, the European Data Protection Directive (DPD) [9], that came into force in 1995, describes the RTBF in Article 12(b) as follows:

> *Member States shall guarantee every data subject the right to obtain from the controller:*
> *(b) as appropriate the rectification, erasure or blocking of data the processing of which does not comply with the provisions of this Directive, in particular because of the incomplete or inaccurate nature of the data;*

In the court case of 'Google Spain v AEPD[8] and Mario Costeja González', Article 12 was used by the Court of Justice of the European Union (CJEU); see section 2.4.1.

[8] Agencia Española de Protección de Datos, the Spanish Data Protection Authority

In the factsheet explaining this ruling on the Right to be Forgotten [10], it is stated that:

> *'In recognising that the right to be forgotten exists, the Court of Justice established a general principle. This principle needs to be updated and clarified for the digital age.'*

2.2.2 The General Data Protection Regulation and Article 17

The updated and clarified version of the principle is described in the upcoming General Data Protection Regulation (GDPR), where the Right to be Forgotten becomes an actual right.

Actually, the GDPR is a modernisation of several pieces of data protection legislation [10], of which Article 17 addresses the 'Right to be forgotten and to erasure'. We refer to Appendix A for the full version van Article 17.

What makes the GDPR probably more effective than current legislation is that Article 3 describes that, *'non-European companies, when offering services to European consumers, must apply European rules.'* In addition, the company has, *'to prove that the data cannot be deleted, because it is still needed or is still relevant'*, and not the individual. The data controller is also obligated to ensure that third parties erase personal data and the GDPR, *'allows data protection authorities to impose fines up to 2% of the annual worldwide turnover'* [10].

2.3 Analysis of Article 17

In March 2014, the GDPR was amended and the title of Article 17 changed from, 'Right to be forgotten and to erasure' to 'Right to erasure'[9]. Article 17, paragraph 1 states that the data subject has the right to obtain, from the data controller, the erasure of his personal data. In other words the data controller has to erase the personal data of the subject if he or she requests this. We now analyse the Right to erasure to deduce on what grounds the data subject can have its personal data erased.

2.3.1 Erasure of personal data

The data subject can request its data to be erased when it is no longer needed for the purpose it was collected, or processed for. We mentioned in chapter 1 that volunteered data requires consent from the data subject, such that the data may be collected and used. When the consent is withdrawn, the data subject has the right to have its data erased. Similarly, the Right to Erasure can be obtained when the storage period, to which the subject consented, expires. Other grounds on which the data subject can obtain the Right to Erasure from the controller are: no legal ground for processing, the subject objects to the processing, the data has been unlawfully processed, or has been made public

[9] http://www.europarl.europa.eu/sides/getDoc.do?type=TA&language=EN&reference=P7-TA-2014-0212

without justification. Additionally, it is possible for a European court or regulatory authority to decide that data should be erased, which also means the subject obtains the Right to Erasure.

If the Right to Erasure is obtained, it means that the personal data cannot be further disseminated. Third parties also have to erase links to, and copies or replications of, the data. The data controller has to verify that the subject actually made the erasure request, and ensure the time limits for erasure and periodic review for need of storage, are observed.

2.3.2 Restriction of processing

Instead of erasure of personal data, paragraph 4 of Article 17 states that processing thereof should be restricted. The restriction means that the data cannot be accessed or processed as normal, and cannot be changed.

The restriction of processing should take place when the accuracy of the data is questioned by the subject, and allows the controller some time to verify the accuracy. Moreover, processing should be restricted when data is no longer needed by the controller to complete its task, but must be maintained for the purpose of proof. Another reason for the restriction is when processing is considered unlawful, but the data subject does not want its data to be erased. Similar to erasure of personal data is that processing should be restricted when a European court or regulatory authority decides so. The same applies when the subject wants to transfer data to a different system, or the storage technology cannot be erased and was in use before the GDPR. If the data has to be erased, the controller is not allowed to otherwise process the data.

2.3.3 The limitations of the Right to be Forgotten

The GDPR also specifies the limitations of the Right to be Forgotten. Meaning that in these areas the RTBF is not applicable [6], because of the importance of the data to the public. The areas include: the protection of the freedom of expression, interests in public health, research for historical, statistical and scientific purposes (Article 17(3)), for compliance with a legal obligation, and in cases where restriction instead of erasure should follow (Article 17(4)) [10], Appendix A.

2.3.4 Observations

From the above mentioned analysis we can observe two approaches of the RTBF that can be applied to personal data: erasure or restriction of processing.

The grounds stated in Article 17 explain why a data subject can request a controller to erase its personal data and in what circumstances the processing of data should be restricted. The article also provides some indication on how this restriction should be executed (i.e. restrict normal access, operations and not change data).

However, there is no indication on how personal data should be erased, which raises questions from a technical perspective. For example, does it mean that the storage medium should be destroyed? Or should the data be removed from a file? In order to translate the legal perspective into a technical one, it is important to better understand the meaning of erasure from the legal perspective. Because we are interested in the situations where the data controller has to actually erase personal data, and in a lesser extent interested in the restrictions on processing, we will focus on only the erasure part of the Right to Erasure.

2.4 Examples of case law

Observing Article 17 alone does not provide us with enough understanding about how erasure of personal data is perceived from a legal perspective. In order to gain more insight in erasure from this perspective, we analyse two cases in this section.

The first case illustrates that personal information in the public domain is accessible through search engines. Which means that also old, outdated information can still be found, and negatively impact a person later in their life. By establishing that search engines are in fact data controllers, the data subject gains the right to obtain the erasure of specific search results from the search engine. The second case does not address the Right to be Forgotten, but does show a different concept to erase data compared to the Google case. Instead of erasing search results, the information is simply not displayed any longer, and consequently removing it for the people who had access, which may include people from the public domain.

2.4.1 Google Spain v AEPD and Mario Costeja González

The current interpretation of the Right to be Forgotten (i.e. Right to Erasure; see section 2.2), as described in the Data Protection Directive, was used in the case of Google Spain v AEPD and Mario Costeja González. We briefly explain the case.

When Mr González's name was searched using Google, the results showed articles about forced property sales due to social security debts. The articles were published in the La Vanguardia newspaper, and on its website at the end of the nineties. The issues described in the articles were resolved, and in 2010 Mr González lodged a complaint against the newspaper and Google Spain.

The complaint concerned that because the issues were resolved, any further reference to the articles was irrelevant. Mr González requested that the newspaper removed the articles, and that Google removed the links concerning him and the articles [10, 11]. The AEPD stated that the information in La Vanguardia was lawfully published, and therefore rejected that part of the complaint.

Eventually the Court of Justice of the European Union (CJEU) ruled that [10]:

- EU legislation applies to search engine operators if they have a branch or subsidiary in a Member State, even if the server doing the processing of data is physically outside of Europe.

- *'Search engines are controllers of personal data'*, which means data protection legislation applies.
- An individual has the right to ask search engines to remove links with personal information. The right applies, *'where the information is inaccurate, inadequate, irrelevant or excessive for the purposes of the data processing'*.
- The Right to be Forgotten is not absolute. Meaning that it needs, *'to be balanced against other fundamental rights'*, and every deletion request has to be evaluated on a case-by-case basis.

However, in his analysis about the case Van Hoboken argues in [12] that the CJEU did:

> *'... not develop an equally thoughtful understanding and appreciation of the positive and essential role that search engines play online'*.

The role referred to is that, search engines make information more accessible to users, and provide an audience for people who want information to be online. Therefore, the removal of links from the search engines goes against the freedom of expression.

One problem, in this case, that was pointed out in the analysis [12], was the addition of "complete" to the fundamental rights jurisprudence.

> *'Simply put, if the protection of privacy and data protection has to be complete in this context, the room for respecting the freedom of expression interests involved may not be optimal'*

For example, one can imagine that "complete" removal of certain links can lead to a form of censorship where information can only be found with very specific keywords or not at all.

According to Van Hoboken [12], a more nuanced judgement is needed, because it is unclear what the responsibilities of search engines are, as this is largely dependent on the context in which the search engine is used.

To clarify the judgement, the Article 29 Working Party (Working Party) provided guidelines on how to implement the judgement in this particular case [13]. Additionally, the Working Party added criteria that can be used by a Data Protection Authority (DPA) to handle new search engine cases.

The guidelines [13] state that a DPA should consider the interest of the public in having access to the information. If it is more important for the public to have access than having the information erased for the data subject, then de-listing[10] will not be appropriate. Furthermore, only searches that include the data subjects name, including variations thereof i.e. family names or different spellings, should be affected. Meaning that the information is still accessible using different (i.e. without a name) search queries, or via the original source. Also, the data subject does not have to contact the original source in order to have its search results removed from search engines. De-listing of search results has to take place on all relevant domains, and not only on domains in the European Union. The search results should not display that they have been modified due to a de-listing request of one particular individual.

[10] De-listing is the term used in the guidelines [13] for removing name related search results.

The criteria to be used by the DPA contain similar views, but also elaborate on the relevance of personal data, stating that the relevance of data, *'is also closely related to the data's age.'* Additionally, they emphasise the importance of whether data is up to date [13]:

> *As a general rule, DPAs will approach this factor with the objective of ensuring that information that is not reasonably current and that has become inaccurate because it is out-of-date is de-listed. Such an assessment will be dependent on the purpose of the original processing.*

The DPA should also consider de-listing appropriate if, *'the data subject consented to the original publication, but later on, is unable to revoke his or her consent, and a delisting request is refused'* [13].

Brief analysis of the Google case We can make a several observations which are of interest to the technical perspective, which follows later in this thesis; see chapter 4.

One observation is that in this case, personal information exists in the public domain, and is made easily accessible through a search engine. Using a casual query on a data subject's name (i.e. just Googling someone), and because it concerns outdated information about a past event, a negative context is created by someone viewing the information in the present. The negative context is harmful to the data subject. Removing the links to the outdated information, from the search results (when a name is in a query), decreases the possibility a negative context can be created by a casual search.

A different view is that the removal of search results requires an increase in effort to find the same casual result, and relies on people not putting in the effort to find it. Therefore, we assume that personal data or information that is not easily found within the public domain, but requires effort, can be considered forgotten. Yet, the personal data still exists in the public domain, therefore it is not explicitly erased from that domain.

Another observation from the Google case is the influence of time on the consideration to erase or keep personal data or information. We will elaborate on this topic in chapter 3.

2.4.2 Europe vs. Facebook

The Right to be Forgotten is not used in the Europe versus Facebook case. However, we analyse this case, because it illustrates a different approach of erasing data. Compared to the Google case, where specific search results are removed, this case explain the erasure of data by not displaying it to anybody that the data was displayed to before.

The case was initiated by Max Schrems[11], who asked Facebook to provide him with all the personal information Facebook had on him. He received a 1222

[11] http://www.viennareview.net/news/special-report/eu-vs-facebook-fighting-for-the-right-to-be-forgotten, accessed on 08-07-2015.

page PDF-document, which also included information of which he thought was deleted. Mr. Schrems decided to file complaints against Facebook Ireland[12] to the Irish Data Protection Commissioner.

In August 2011, the first 22 complaints were filed[13] and covered subjects such as: not deleting certain data, shadow profiling, face recognition and excessive processing.

Additional complaints were filed in September 2011, which included subjects such as: deleting links to pictures, tracking users via the like-button, and frequently changing policies.

One of the main issues was that data, for example pokes, posts, friends, and chat messages, were not deleted by Facebook even though the user did click on 'delete'. Instead of erasing the data, Facebook set the data to 'not-to-be-displayed', which led the user to believe that the data had actually been erased. Pictures were also not deleted directly, only the links to the pictures were deleted. According to the complaint, the deleted pictures were still available for several hours after the user 'deleted' them.

After years of legal conflicts, the procedure ended in July 2014 with the decision by the Max Schrems team, to withdraw the initial 22 complaints.

> *This decision was based on the fact that the Irish DPC has refused a formal decision for years and has not even granted the most basic procedural rights (access to files, evidence or the counterarguments). The DPC has factually stopped all forms of communication and ignored all submissions made. Many observers assumed that this may be based on political and economic considerations in Ireland.*[14]

Following the withdrawal of the complaints, a class action lawsuit was started against Facebook in August 2014 with the aim, *'to make Facebook finally operate lawfully in the area of data protection*[15] *'.* However, the complaints in this specific case are less precise and instead focus on:

- *'Data use policy which is invalid under EU law;'*
- *'The absence of effective consent to many types of data use;'*
- *'Support of the NSA's 'PRISM' surveillance programme*[16]*;'*
- *'Tracking of Internet users on external websites (e.g. through 'Like buttons');'*
- *'Monitoring and analysis of users through 'big data' systems;'*

[12] A Facebook subsidiary concerned with users outside the United States and Canada.

[13] http://www.europe-v-facebook.org/EN/Complaints/complaints.html, accessed on 08-12-2014.

[14] http://www.europe-v-facebook.org/EN/Complaints/complaints.html, accessed on 08-12-2014.

[15] http://europe-v-facebook.org/EN/Complaints/Class_Action/class_action.html, accessed 08-12-2014.

[16] A top secret program allowing the NSA access to data from Google, Facebook, Apple and other major IT-companies.

- *'Unlawful introduction of 'Graph Search'*[17]*;'*
- *'Unauthorised passing on of user data to external applications.'*

On the 1st of July 2015 the court Vienna rejected the case on procedural grounds, because Max Schrems used Facebook *'for commercial promotions of his publications'*[18]. The court forwarded the case to a higher tribunal, and Max Schrems said that he would appeal the decision. At the moment of writing the lawsuit is continuing, and thus no conclusive results have been published.

Brief analysis of the Facebook case As with the Google case, we can make several observations which are of interest to the technical perspective. Facebook shows two approaches to erase data, that are different from the Google case: not displaying data, and deleting links to a file; i.e. a picture.

By not displaying the data, the Facebook-user assumes that the data is deleted, because he cannot view or access it any longer. The same applies to everybody who had access to the data, e.g. the user's contacts (i.e. Facebook friends) or, if it was a public profile, people from the public domain. However, Facebook still stores and has access to the data, as such data is not explicitly erased. Thus, from the point of view of the people who had access at some point (i.e. access domain), the data seems to be erased. Although it is more applicable to consider it forgotten from the access domain than deleted.

Access to data can also be denied, by deleting a link to a file, which is similar to the Google case. Yet, in the Facebook case it is not possible to use a different method (e.g. different search queries) to still access the data.

In contrast to the Google case, this case revolves less around information being deleted from the public domain, but about deleting it from a specific environment (i.e. Facebook). The method of not displaying 'deleted' data works, because the specific environment has control over who has access to the data. Google does not have control over who specifically has access, but only over the ease with which information can be found.

2.5 Initial concept of the legal perspective

Using the analyses of the Facebook and Google case, discussed in the previous section 2.4, we can derive an indication where an intervention can take place, such that personal data can be erased or forgotten. We expand this initial concept over the next chapters with other concepts, and use it to construct a model which can be used to form a technical perspective; see section 4.

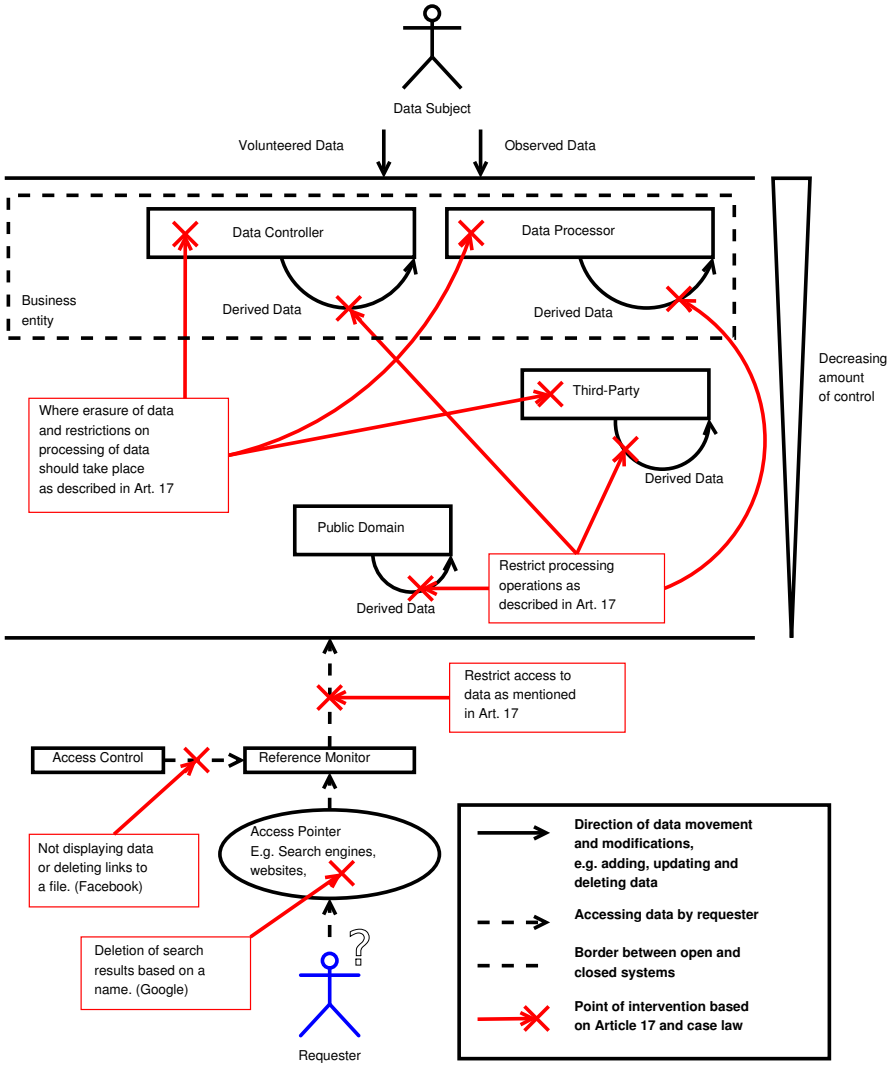

Figure 1: Initial concept with identification of discussed methods to erase or forget personal data

2.5.1 Model of the initial concept

Figure 1 shows the different types of data as discussed in chapter 1. The arrow on the right, marked 'Decreasing amount of control', illustrates the loss of control over personal data by the data subject. The further data travels away from the subject, the less control the data subject has over its personal data. If personal data is publicly accessible (i.e. it resides in the public domain), all control is lost. It is not difficult to imagine why: if information can be accessed by anyone or anything, it can be copied, altered, and republished; see chapter 1.

The data subject supplies volunteered and observed data[19] (top of Figure 1) directly to the public domain, or it can end up in the public domain via any of the other stakeholders: controller, processor or third-party. These three stakeholders and the public domain can also exchange personal data amongst each other. For example, a data controller can access information from the public domain, pass it on to a processor, who derives more or different information from the data before sending it back to the controller. The data controller can also process the data within its organisation, or pass it on to a processor, or a third party. The data controller and processor also analyse, combine or mine the volunteered and observed data. For example, to create a profile of the subject that suits their business model; see the circular arrow marked 'Derived Data' in the figure.

We mentioned in section 2.1 that role of controller and processor can be fulfilled by the same business entity. This is shown in Figure 1 by the dotted rectangle.

The 'Requester' represents anyone or anything that has access to the public domain, or the business entity; see Figure 1. For example, a requester can be someone browsing the WWW using a search engine, or a search engine indexing information from the public domain, or someone using a social network. The access pointer is an object which contains information that points to the location of certain data; e.g a search engine. The reference monitor, in combination with access control, validates if access to the personal data is allowed or not.

The observations made, following the analysis of Article 17, in section 2.3 describe that there is no indication on how to erase personal data. Based on Article 17 alone, we can only indicate where Article 17 has an effect, if a data subject decides to invoke its Right to Erasure, which is marked by the red arrows pointing to the controller, processor and third-party.

Similarly we have indicated where restriction of processing should take place. One way to restrict processing is to prevent access to the personal data, e.g. indicated by the red cross on the arrow from the reference monitor. Restriction on processing operations is indicated, for example, by the red arrows pointing towards the derived data arrows, because derived data is the result of processing.

From the analyses of the Google and Facebook cases in section 2.4 we observed two methods on how personal data can be forgotten from the public domain:

[17] A search engine that makes it possible to find information about Facebook-users more easily.

[18] http://uk.reuters.com/article/2015/07/01/us-facebook-austria-lawsuit-idUKKCN0PB42O20150701, accessed on 07-07-2015.

[19] Note that the data subject is not always aware of observed data.

increasing the effort needed to find, and denying access to, personal data. The Google case shows that removing specific search results (i.e. results of name based queries), derives in an increase of effort needed from the Requester of the search engine to find a similar result. In turn the Requester gives up, which results in the forgetting of personal data, but not the erasure thereof. This is indicated by the red arrow, pointing to the access pointer.

For example, users use a search engine to find information, unless a website that holds the information is already known to the user. When a search engine does not reply with a satisfactory result, the user could try a different search query, or give up searching all together. In the latter case, the information would not reach the user, therefore not be processed or digested, and thus forgotten; but not erased. Similarly, a search engine can suppress a search result by making it not a high-ranking result, but place it further down in the results; making it less likely a user finds it.

The Facebook case shows a different approach. By not displaying information to users of system, to which the information was shown before, access to the data is denied. Therefore, the information will be forgotten within the group (that previously had access), which includes users who have access using the public domain (e.g. someone viewing a public profile), or other users of the system (e.g. other Facebook users). This method of forgetting is indicated by the red cross at the access control, in Figure 1.

To summarise The grounds stated in Article 17 indicate *why* a data subject can request a controller to erase its personal data. Our analysis of the case law examples show that there are at least two methods which could be used to forget data: increasing the effort to find, and denying access to, information. Yet, these methods do not explicitly erase personal data, and therefore it still persists.

This raises the question: when is data or information actually forgotten or erased? To answer this question, we have to explore the concept of forgetting.

3 The Socio-philosophical Perspective On Forgetting

Our observations, from the previous chapter, regarding Article 17 show that the grounds state *why* a data subject can request a controller to erase its personal data, but that the article does not explain *how* the erasure of personal data should be carried out. The two legal cases which we discussed provide two methods on how erasure of personal data could be achieved. However, the analysis of the legal cases show that personal data is not explicitly erased, it is merely 'forgotten' by not being accessible, or by being difficult to find. Which raises the question: when is data or information actually forgotten or erased?

In this chapter we analyse the concept of forgetting, such that we can better understand when data can be considered forgotten or erased. To achieve this we first describe responses against storing digital information for an indefinite period of time, and loss of power and control over the data. Then, we briefly explore how humans forget information, and elaborate on the factors that contribute to forgetting.

3.1 Responses

In the introduction, see chapter 1, we mentioned that technological advances made it possible that an increasing amount of data is stored in digital form; for an indefinite period of time; and that the data subject loses power and control over the data [2].

In his book about forgetting information in the digital age, Viktor Mayer-Schönberger discusses six potential responses [2]. The first three responses are relational, because they address the movement of data. *'They presume that the sender decides whether and under what conditions the recipient should use the information'* [2]. As such there is a relation between the sender and recipient of the personal data. The last three response are more speculative.

We summarise the response as follows:

- Digital abstinence: where people stop sharing information. This can be difficult to achieve, because people like to share information. However, even information that is shared under specific conditions, is still beyond the control of the owner. Digital abstinence could also be considered as a form of self-censorship. Moreover, when individuals become digitally abstinent, it does not mean that data controllers and processors become abstinent too. In order to achieve that, information privacy rights are necessary.
- Information privacy rights: *'... give individuals a right to choose whether or not to share information'*. Violations of those rights, result in punishment for the violator. For example, the articles in the GDPR, and more specifically Article 17, the Right to Erasure; see chapter 2. In the situation that privacy rights are abolished, or not enforced, the previously

collected data (now lacking protection) can be used without any restrictions. Therefore, it is argued in [2], the rights need to be future proof. Several concepts have been proposed to mitigate this problem, and make privacy rights more effective; see the information ecology explanation below. However, the data subject does have to put in effort (i.e. court cases) to challenge violations of their privacy rights, and remain in control over their information. To reduce this effort, and still regulate behaviour, enforcement needs to be simplified, which could be achieved through technical alternatives instead of the conventional legal mechanisms [2].

- Digital privacy rights infrastructure: similar to a Digital Rights Management (DRM) system, it allows users more control over their data. However, the rules necessary to specify what is and is not allowed to be done with personal data, are difficult to draft and translate to a technical system.

These three responses are not able to handle the loss of control (power) over data, and the effect of time on data. The challenge with time, according to [2], is that too much information, which builds up over time, will negatively affect *'our abilities in the present.'*

To overcome these issues, different responses are necessary. The following responses address the concept of digital memory: *'...a set of information stored on digital media and thus frozen in time...'* [2].

- Cognitive adjustment, where people *'consciously disregard old facts and accept that humans are ever-changing beings, never locked into a particular state'* [2]. It is unlikely that the adjustment will happen suddenly, but rather take a very long time.

  Moreover, the data that has been stored is incomplete, because it is the result of specific processes (i.e. thought process) that preceded it. As such, the context of the information is missing, and in combination with the large quantity of information available, makes making the decision to disregard old facts difficult. It is suggested in [2] that what is stored should, therefore, be regulated.

- Information ecology: *'a deliberate regulatory constraint of what information can be collected, stored, ..., by whom and for how long.'* This is part of information privacy rights, and aims to prevent the use of data in a unfavourable manner in the future. However, in certain circumstances these constraints are negated or made less effective, for example by the use of data retention legislation.

- Perfect contextualisation: a concept that would require to store not only data, but also store the context surrounding the data. This would require far more information to be stored, which then has to be interpreted, and help people evaluate certain information in its specific context (as opposed to data viewed out of context). One disadvantage, similar to cognitive adjustment, is the incompleteness of the data. Furthermore, the amount of information required to view something in its complete context is immense, and will probably result in an uncomprehensive understanding. The main reason for this is the effect time has on human interpretation, because

humans look at a past event with a different perspective than when they experience the event in the present.

3.2 The concept of forgetting

To better understand the concept of forgetting we briefly explore how a human brain forgets information, and what tools we use to help us remember.

3.2.1 Forgetting in the human brain

In a paper [6] that studies time as a factor which decides whether data should be retained or forgotten, Korenhof et al. describe forgetting as *"a "fail[ure] to remember", a glitch somewhere in the memory process that either temporarily or permanently fails to retrieve specific information."*

There are three factors that play a role when something is forgotten in the human brain [6]:

- *"the passing of time"*: new information is more persistent in memory than older information;
- *"the meaning of the information"*: meaningful information is more persistent than meaningless information;
- *"the regularity with which the information is used"*: regularly used information is more persistent than irregularly used information;

3.2.2 External transactive memory

To overcome the problem of forgetting and help with remembering, humans have used tools; e.g. agendas, books, photographs, films, etc. These tools are categorised as external memory, because information is stored outside of the brain. However, due to storage space limitations, the tools still retains a form of forgetting known as, forgetting-by-selection. For instance, when storage space runs out, a decision needs to be made of what data to keep (i.e. remember) and remove (i.e. forget) [6].

The technological advances, mentioned in chapter 1, provided humans with an external digital memory, which helped overcome the problem of forgetting-by-selection. In turn, this led to a new norm: remembering-by-default.

An extended form of the external digital memory, is known as a transactive memory. A term used by Korenhof et al. in their model of digital memory [6]. A transactive memory has the property that everyone who is connected to it, can also add data to the memory and retrieve data from it. An example of a transactive memory is the World Wide Web (WWW).

3.3 Forgetting in an external transactive memory

We have established that an external transactive memory is a form of external digital memory where everyone connected to it can add, or retrieve data. In this section we investigate how personal data can be forgotten from a transactive memory. To achieve this we have to translate the concept of the transactive memory to a technical model.

3.3.1 Open and closed system

We use a report by ENISA[20], about the technical means that support or enforce the Right to be Forgotten in information systems, to make an initial distinction between systems involved with a transactive memory.

The ENISA report differentiates between an open system and closed system [14]. In an open system, personal data can be copied by anyone, and no means are in place to keep track of these copies. The report states it is impossible to enforce the Right to be Forgotten, using technical means, in an open system [14], because:

> *In such an open system it is not generally possible for a person to locate all personal data items (exact or derived) stored about them; it is difficult to determine whether a person has the right to request removal of a particular data item; nor does any single person or entity have the authority or jurisdiction to effect the deletion of all copies.*

In a closed system,

> *'all components that process, transmit or store personal information, as well as all users and operators with access to personal information can be trusted or held accountable for respecting applicable laws and regulations concerning the use of the private information.'*

An example of a closed system is an Electronic Patient Record (EPR), where patient data is shared among caregivers. In this case it is possible to use technical means to enforce the Right to be Forgotten [14].

Both open and closed systems can be seen as transactive memories, because multiple parties can add and retrieve data from the systems. However, we find that a nuance is necessary. Both systems usually offer an interface which allows data requesters to interact with them. With an open system this interface is publicly accessible, which indeed makes it difficult to locate and erase personal data; as stated above. The interface does, however, create a checkpoint where certain conditions can be applied. The Google case illustrates this, as the search engine can be seen as the interface to which certain conditions can be applied, e.g. not showing specific results.

[20] *'The European Network and Information Security Agency (ENISA) is a centre of network and information security expertise for the EU, its member states, the private sector and Europe's citizens.'* [14]

3.3.2 Examples of forgetting in a transactive memory

We relate the two methods of forgetting from the previous chapter (increasing effort to find data and not displaying data) to the concept of transactive memory.

Removing specific search results increases the effort needed to find data; see chapter 2. However, data can still be found if the effort is increased, e.g. by using a general term and inspecting more search results. This initially divides the group we classified as requesters in our initial concept, see the previous chapter, into two groups: an easy-access group, and a put-in-more-effort group. From the point of view of the easy-access group, the data does indeed seem erased, but not from the perspective of the put-in-more-effort group; they still find the data.

Not displaying data to a specific group introduces a similar situation. Data that was visible to a group, suddenly becomes invisible. Thus from the perspective of that group, data does indeed seem deleted. Yet, to a different group that can still view the data, it is not deleted.

From these two methods we deduce that forgetting, or erasing data in a transactive memory, such as the WWW, is not equal across all concerned parties. In fact specific parties can still use the data, while others cannot. From this observation we deduce that forgetting data can be considered as: denying a specific group, access to data. We clarify this idea in the next section.

3.4 Model of forgetting in an external transactive memory

In this section we clarify the concept that forgetting in a transactive memory is unequal across the concerned parties, because a specific party, or group, is denied access to data. We illustrated this inequality in Figure 2 by the use of domains, which we explain below.

Personal data itself resides on a storage medium, which is part of a system. The system domain is the first layer around personal data that has access to it, and can include multiple systems of the same organisation.

Systems that store and process personal data are part of an organisation, as such the organisation is the second layer to have access to data. Organisations that operate in the (semi-)public domain are on the 'Open system' side of the dividing line. On the 'Closed system' side are organisations that do not operate in the public domain.

The division between semi-transactive and transactive memory domain is necessary to make a distinction between organisations that offer a service to the public, but require some form of membership to use that service. For example, free email (e.g Gmail) or a social network (e.g. Facebook).

A method that affects the interface (i.e. access pointer), e.g. deleting specific search results, only affects arbitrary requesters, or requesters that are marginally aware of the existence of particular data. Arbitrary requesters access personal data at random, e.g. through an arbitrary search query. The requester is marginally aware of the existence of data when it became aware of it through a

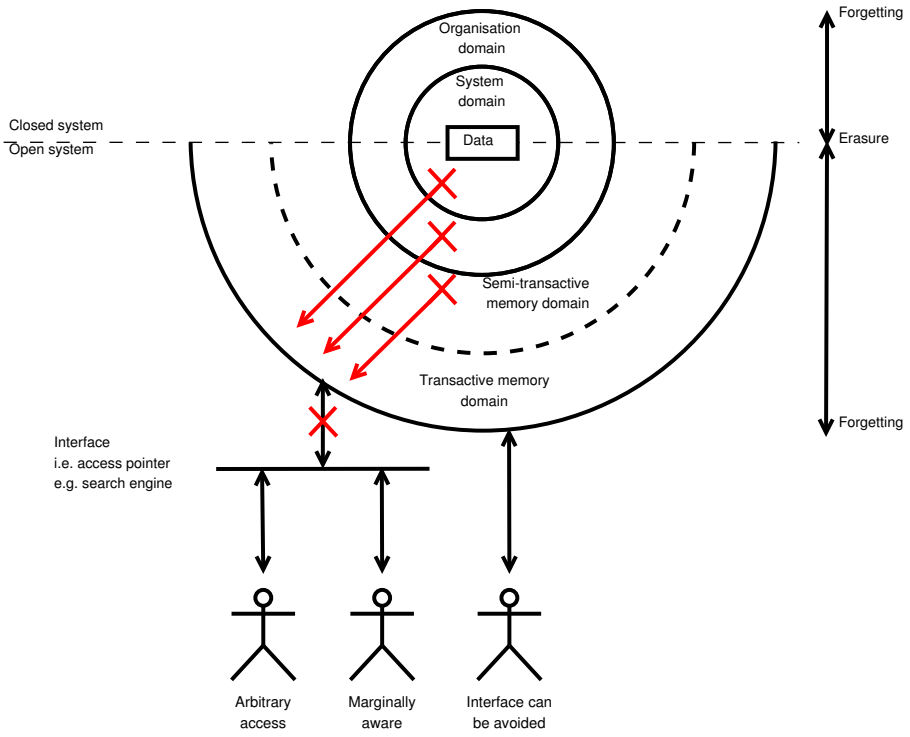

Figure 2: Effect of forgetting or erasing in a transactive memory across different domains.

different channel, but uses an interface to gain access again. For example, when somebody forgot a link to a specific website, and uses a search engine to find it again. The interface can be avoided, when a requester already knows where the data can be found.

The effect and inequality of forgetting methods is indicated by the red arrows; see Figure 2. The closer a method is applied to the actual storage medium, the more domains are affected. For example, a system cannot access personal data once it has been removed from the storage medium, whereby an organisation cannot use it, or make it accessible, as such it cannot be accessed from the semi-transactive, or transactive, memory domain. A method, e.g. not displaying data, that is deployed at semi-transactive domain level also affects the transactive domain, but the organisation and systems could still use the data.

3.4.1 Difference between forgetting and erasing data

As we mentioned the closer a method of forgetting takes place to the actual storage location of the data, the more domains will be affected. In this situation we consider that the method actually erases data. Methods that are deployed on a domain further away from the data, do not erase data from a system, but merely deny access to specific parties.

3.5 Factors that contribute to forgetting

In section 3.2 we described three factors that contribute to forgetting in the human brain. Forgetting, or a failure to remember, happens automatically or passively; it is not possible for humans to actively forget something. On the contrary, an external transactive memory (i.e. the World Wide Web) does not forget (i.e. erase or delete) data passively, but remembers (i.e. stores) by default. Such a transactive memory only erases data when it is actively told to do so.

To determine if personal data needs to be deleted from a transactive memory, several factors can be used. In this section we will investigate these factors and place them in a technical context.

3.5.1 Time

Time is one of the factors that contribute to forgetting in the human brain, but time also influences the significance of information. The concept of time is also reflected in the legislation discussed in chapter 2. The google case, discussed in section 2.4.1, is an example of what role the concept of time can play in law. In this case, links kept referring to social security issues which were resolved some time ago.

The roles that time can play in law have been discussed by Korenhof et al. [6]:

> *"On the one hand, 'time' can play a role as a weight in a balance of interests, as a factor adding or removing weight to the request to 'forget' personal information or its opposing interest, resulting in tipping the balance in either direction. On the other hand, 'time' can play a role as the marker of a discrete moment where the grounds for retention no longer hold and 'forgetting' of the data should follow."*

In the case of balancing interests, a careful balancing of the interests for all involved parties is needed, in order to determine if data should be retained or forgotten. It is stated in [6], that in this case, it needs to be recognised *'that information has a lifecycle and its value (also to the different interested parties) changes over time.'* Which makes using 'time' to balance interests, complex. For example [6]:

> *'A specific time span can mean something completely different for the data subject (lifetime perspective), the data controller (processing and use time) and for third parties (public interest, transactive memory use). The passing of ten years in time has a different meaning in relation to the lifetime of an individual than it has in relation to historical interest of the public.'*

The case where time is used as a marker, after which retention no longer holds and forgetting should follow, is less complex. Examples of such moments in time are, the end of a data retention period, or the purpose for which the data was initially collected, was reached.

In either case, there is a point in 'time' where the need for the individual to be forgotten outweighs the need to store the information.

The second case (time as a marker) is easier to implement in a technical system than to use 'time' alone to balance interest. For example, we could use time as a marker to actively tell a system that specific information should be deleted at that moment in time. In order to balance interests, however, time alone is not enough: the meaning of the information needs to be known.

3.5.2 Meaning of information

For humans, meaningful information is more persistent than meaningless information. This persistence shows that the meaning of information and time are dependent on each other. Meaningless information, for example, will be forgotten more quickly than meaningful information. Additionally, the meaningfulness of information could decrease over time, but in certain circumstances (i.e. events of historical importance) it could increase. The dependency on time, and meaning of the personal information, makes balancing the interests to erase, or not to erase, personal data difficult.

However, the meaningfulness of information also differs per individual; information can be meaningful to one person in particular, while completely meaningless to the other.

The systems used by data controllers and processors cannot measure the meaningfulness of a single piece data. Using algorithms, the systems have to be told (i.e. programmed) what data is significant, relevant, important, and therefore that it has meaning; it is not told what that meaning is. In other words, the systems are programmed to distinguish what is significant, depending on the context in which data was gathered. Only when separate pieces of data become connected with each other, through the algorithm, it gains in significance, such that something can be said about the data. When something can be said about the data, it becomes information.

3.5.3 Regularity

Another factor that contributes to forgetting is the regularity with which information is used. More regularly used information is more persistent than less regularly used information.

In an external transactive memory, the regularity can be used as an indicator to decide whether data should be removed or not, which can be achieved in combination with time.

Data that is used regularly, within a certain time period, can indicate that it is more significant than data that is used less regularly in the same time period. A decrease in regular use over time can also indicate a loss of significance.

On the other hand, it is possible that the regularity of use decreases over time, but the significance of the data increases. An example of this, is the historical importance of the data.

Another measurable factor that is related to the regularity and significance, is the amount of parties using the data. If a multiple parties use the data, it could

indicate that the data is significant, because the regularity of use increases. If less parties use the data, the regularity drops and thus the significance of the data.

However, this still depends on the importance and relevance of the involved parties. An important and relevant party will increase the significance of the data, even though the regularity with which the data is used is lower.

3.5.4 Space

In addition to these three factors, the concept of space as a means to determine if data should be erased, is also mentioned in [6]. This seems counter intuitive, because the amount of stored data, and the storage capacity of devices is increasing.

For example, early digital devices had a very limited storage capacity, and users had to determine what data was significant enough to store or insignificant enough to erase. Although, as we have seen in chapter 1, the concept forgetting-by-selection has been replaced by remembering-by-default.

However, reintroducing a limit of storage capacity on devices, or allowing only a maximum storage capacity to be used, may not be favourable option for the storage medium industry.

Similar to limiting storage capacity, is posing a maximum on the amount of personal data points that can be stored per person, per party. The party would have to make a decision about what data to keep and what data to erase. In this manner only the most significant data, for that particular party, is stored.

It could also be used as a moment in time to notify the data subject. The subject would receive a message that a certain amount of data has been collected. After which the data subject would request the controller or processor to erase specific personal data points.

4 The Technological Perspective Of The Right To Be Forgotten

In chapter 2, about the legal perspective, we discussed two cases that both illustrated methods of how personal data can be forgotten. Yet, we noted that these methods do not explicitly erase data. The socio-philosophical perspective, as discussed in chapter 3, further explored what forgetting actually means, what factors contribute to forgetting, and that the location where forgetting takes place has a different impact concerning different domains.

Based on chapters 2 and 3 we regard forgetting data as, a failure to access data by a certain requester that wants access to the data. In this chapter we refine this concept for the technical perspective, and indicate where accessing data can be obstructed, such that it can be considered forgotten or erased.

4.1 Model of forgetting by denying access

In chapter 3 we explained that personal data can be accessible from different domains, and that forgetting can be regarded as a failure to access that personal data. In order to implement forgetting, we need to describe how data is accessed and information obtained from a technical perspective. This is visualised in Figure 3. The figure[21] describes how a requester can gain data access, and where access by that requester can be obstructed (which will be explained later in this chapter).

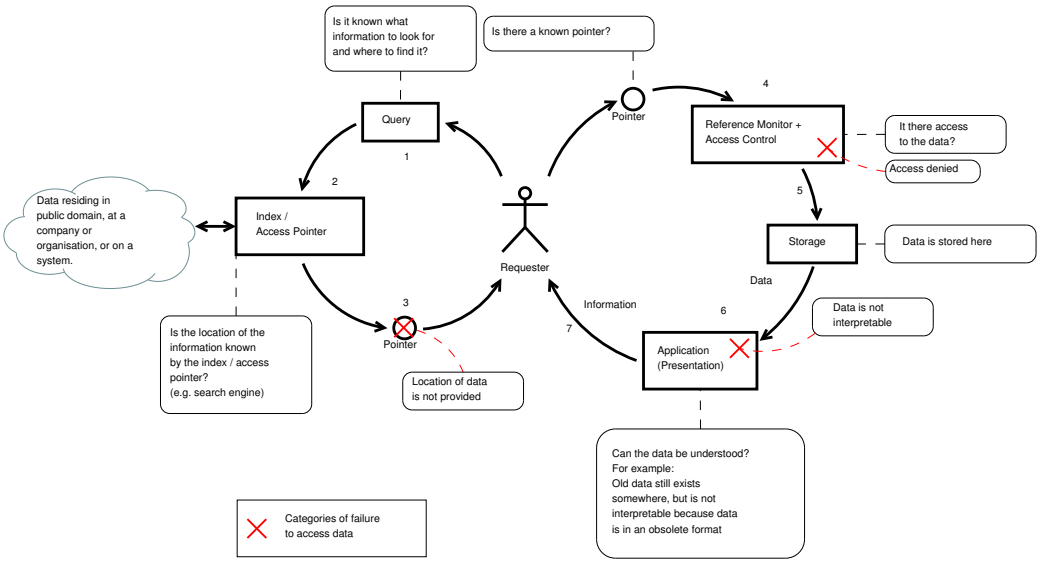

Figure 3: Forgetting or erasing personal data by denying access

[21] Note that we abstract from the separate roles of controller and processor, and consider them the same entity (e.g. company or organisation); as discussed in chapter 2.

The requester in the middle of Figure 3 has two options: it wants to know something or it already knows where to find certain information. If the requester wants to know something, it has to know what information to look for, and where there might be a possibility to find it. For example, by forming a search query and using a search engine to find the information.

The query (1) posed to the index (2) will result in a pointer that provides a location of the actual data (3). It is not possible to supply a pointer if, the answer to the query is not known to the index, or it is not allowed to provide a specific result (e.g. the Google case; see chapter 2).

Once the requester has obtained a pointer, it can request access to the data using the reference monitor (4). In turn, the reference monitor has two possibilities: either deny or allow access. If the requester is allowed access, it can retrieve the data, after which an application has to process the information. In the case that accessed is denied, data cannot be processed, and can therefore be considered forgotten or erased.

When data is allowed to be processed (5), the application turns data into information[22] that can be used by the requester. However, in certain situations data cannot be interpreted, because e.g. the data is damaged, or in an obsolete format (6).

In the final step the requester receives the information from the application, and decides whether it is useful or different data is needed (7).

4.2 Applicability of the model regarding a requester

The model illustrated in Figure 3 can be applied in different domains, depending in which domain the requester resides. For example, the requester can be:

- a system which gets an assignment to retrieve data from somewhere else than itself;
- a system which gets an assignment (e.g. from software) to retrieve data from within itself;
- an organisation that wants information from somewhere else than itself;
- an organisation that wants information from within itself;
- an individual that wants to find certain information.

The model can be applied in all of the domains from chapter 3. To clarify we explain one example: if a system wants information from within itself, it has to know a pointer to the file that holds the requested data. In case that data is corrupted or damaged, the application cannot interpret the data, thus failing to access data successfully, which results in losing the information.

[22]Note that the processed data now has a purpose for the entity, which means it is information instead of data

4.2.1 Efficacy of forgetting or erasing in the model

We showed in chapter 3 that personal data can be accessible through different domains, and that there exist several methods which remove data from these domains. Although it results in information being forgotten, it is not forgotten equally over all involved domains. This creates a difference between forgetting and erasing data. As we have explained in chapter 3, the closer a method is applied to the system domain, the more domains lose access to the data, which means data is erased. While methods applied further from the system domain, result in data being forgotten.

For example, if the physical medium (e.g. a hard drive), on which the personal data resides, is destroyed it becomes impossible to access that data from a different domain. Which results in the personal data being erased from any domain it could be accessed from.

4.3 General methods of erasing or forgetting

We mentioned that erasure or forgetting of data can be considered as a failure to access data. We use the model to identify general methods that let an entity deliberately fail to access data.

There are three general methods, which are indicated in Figure 3 by red crosses. The methods are as follows:

- Denying access to data: a requester does not have data access
- Withholding the location of data: a requester cannot access the data without the right location
- Rendering data uninterpretable: the application cannot interpret the data

Any of the general methods will result in personal data not being accessed successfully, and thus cannot be used to form information with the result that it is considered erased or forgotten. As mentioned in the previous section, the model can be applied across the different requesters, domains or situations. Which means that we can use the general methods as categories, such that we can further define methods of forgetting or erasing independent of the entities, domains and situations. In other words, we can divide the methods of forgetting and erasing into three categories per domain, requester or situation.

4.3.1 Subdivision of methods of erasing or forgetting

The above mentioned methods are too unspecific to from a framework, which can be applied to technology. In order to make a specific framework we divide the main categories, from the previous section, further into subcategories as follows:

- Denying access to data:
  - Change permissions to read, or write data
  - Not providing or retracting the required credentials to access the data
- Withholding location of data:
  - Modifying results depending on the query
  - Removal of pointers to storage locations
  - Moving of data to different storage locations
- Rendering data uninterpretable:
  - Destruction of the storage medium
  - Data is damaged
  - Data is transformed deliberately

4.3.2 Applying the framework to known cases

The Google and Facebook cases, which we explained in chapter 2, introduce two methods of forgetting: increasing effort to find data, and not displaying data. We explain the applicability of the above mentioned framework by classifying these methods of forgetting. First we decide in which domain the method is applicable.

For the Google-case: the domain is focussed on easy-access, or the easy-access-domain, as explained in section 3. The method to forget was, not to reply specific results that contained a name and a search term, which resulted in the data subject not being linked to an event in the past and currently irrelevant. However, the data is still accessible if a query without a name is used. Thus, the results are modified depending on the query, which fits them into that subcategory and corresponding category of 'withholding location of data'. Which makes sense, because the location (i.e. search result) of the data is not provided.

The Facebook-case focusses on the organisational domain, because the data is not displayed any longer to entities that previously could see the data, except for Facebook; which does still have access. In other words, entities that previously had access, no longer do, such that the data seems erased. The not-displaying-data method can therefore be placed in the subcategory of 'change permissions to read, or write data' and corresponding category of 'denying access to data'.

5 Classification of technology

In this chapter we investigate implementations of erasing that are applicable to the system domain, because those implementations influence the most other domains. As there are many ways in which data can be removed from a system, we group the implementations and concepts according to the technology used to erase data (e.g. cryptography). We classify the implementations based on the subdivision of methods of erasing or forgetting, see 4.3.1, and investigate if any implementations rely on the factors from chapter 3, to decide if data should be erased.

The ENISA report [14] mentions several implementations, which support the Right to be Forgotten in information systems. The implementations are related to Viktor Mayer-Schönberger's concept of tagging personal data with an expiration date. In this concept, information systems would have to check the date and erase its corresponding data accordingly. However, this concept is difficult to achieve as it is, against the interests of business and government, technically difficult to verify that systems actually delete data and hard to hold offenders accountable. The research projects discussed in the ENISA report address various aspects of these difficulties [14], and summarise them below. We also included concepts that we have found independently of the ENISA report. The discussed methods are by no means all possible ways to delete data, as there are many ways to achieve this.

5.1 Cryptography

One way to make the erasure of personal data more manageable, is to encrypt the data and when it needs to be deleted, delete the key instead of the data. In other words, it is easier to delete keys because they are smaller and more manageable than to delete the data that is often larger in size. As the data is inaccessible until the encryption scheme is broken, or the key can be recovered using a brute-force attack, the data cannot be used by anyone. Therefore the data can be considered as deleted or forgotten.

In the lines of Viktor Mayer-Schönberger's concept, the encryption key can be associated with an expiration date, which allows files to be erased after a specified period. The concepts described below, use this property.

5.1.1 Revocable backup system

In 1996 a system was introduced that used cryptography to remove files from a system and its copies from the backup tapes. Before a backup on a tape is made the file is encrypted using a randomly generated key. If the file has to be deleted, the user instructs *"the system to 'forget' the key used to encrypt the file"* [15]. In the system, described in [15], the user can specify the files that have to be encrypted when backed up. Each file is associated with an encryption key and each encryption key has a life time specified by the user. The key management works as follows [15]:

> *"When the key expires the system generates a new key for the file and adds the expired key to a list of old keys. The maximum number of expired keys that the system remembers is a parameter specified by the user. When the list of keys is at its maximal size the oldest expired key in the list is erased to make room for the new key."*

For further details regarding the key management of the system we refer to [15].

5.1.2 Ephemerizer

The Ephemerizer is a concept which is similar to the one mentioned above, and introduced by R. Perlman in [16]. The main difference with the revocable backup system, is that also local storage is protected and the interaction with key manager is more secure.

The Ephemerizer concept allows the expiration dates to be set to a day, for up to 30 years, which means about 10000 keys are necessary to cover all days [16]. As it is difficult for a user to manage all the keys and to make sure that there are no copies of the keys, a tamper-resistant smartcard should be used. Key generation and cryptography take place on the smart card. The paper mentions that ephemeral keys (i.e. encryption keys) are used to encrypt files, and 'ephemerizers' (i.e. key managers) are needed to decrypt them [16].

Furthermore, the paper describes two designs. In one design the expiration time is known at the time of file creation. The ephemerizer(s) have a long term key, which is used to certify the ephemeral key pairs (e.g. 10000). The public key of the key pair corresponds to an expiration date, and is used to encrypt a file on the client side. *'This means that the message is secret key encrypted with some secret key K, and K is encrypted P. So the client has* $\{K\}P$, $\{msg\}K$*'* [16].

It creates the ephemeral ciphertext, which is encrypted again with the client's long term key, in order to store it securely on backup media. The file can be obtained by first removing the client's encryption layer, thus decrypting it using client's long term key, which again obtains the ephemeral ciphertext. To decrypt ephemeral ciphertext the client communicates with the ephemerizer using a blinding algorithm.

The client creates a blinding[23] function pair B and U, and sends B({K}P) to the ephemerizer. In turn, the ephemerizer applies the ephemeral private key to B({K}P), and returns B(K) [16]. The client unblinds using U to obtain K, and can now decrypt the message.

The second design uses the ephemeral keys to allow for files to be deleted on demand. In this design the user can decide to delete a file at any given moment, in contrast to setting an expiration date and letting the file be erased automatically. We will not further describe this design, because it is less interesting in the context of this thesis. For details we refer to [16].

A paper by Nair et al. citehybrid describes a flaw in the triple encryption scheme used in original Ephemerizer system of R. Perlman [16]. Using an oracle attack

[23] Blinding allows the ephemerizer to decrypt the encryption key without knowing the input and output of the function; see [16] for details.

it is possible to obtain encrypted information that is not linked to an ephemeral key. This means that the encrypted data can be stored indefinitely, providing the attacker with an indefinite amount of time to obtain the data. An adapted Ephemerizer scheme is proposed using Identity Based Public Key Cryptography (IB-PKC), which mitigates this problem [17].

5.2 Overwriting

A paper by Joukov et al. [18] introduces several problems regarding the secure deletion of data from storage media. It states that deleting data, or emptying the trash bin, does not really delete data and is still recoverable, especially when data has not been overwritten. To mitigate this problem several solutions are mentioned: overwrite data, destroy the physical media, or encrypt the data and securely dispose of the key.

However, encryption brings additional costs e.g. CPU overhead, and requires proper key management to prevent keys from being lost, or broken. Moreover, old encrypted data can still be recovered if not overwritten and be decrypted with an old, compromised key. Still, this is feasible for organisations who already use cryptography, but not for individuals or smaller organisations.

According to [18], data overwriting has its problems too. Data overwriting tools are not reliable, or have performance issues, which discourages user from utilising them. Also, meta-data of files, e.g. file name, size, owner, etc., is not overwritten, but ignored; thus still providing information. With the introduction of versioning file systems, several versions of a file are stored, and not all are overwritten. Storage media that have caching features actually prevent multiple overwrites, in order to store all different operations and write all of them in one time.

If data is overwritten, even multiple times, it is still possible to recover it. However, the paper states that *'... for most users, a single overwrite will suffice and greatly enhance security. In particular, one overwrite will make any software-based data recovery impossible.'* [18]. Although, hardware based recovery is still possible.

In [18] two methods have been described that allow for more secure deletion in a file system (i.e. ext3). The first method relies on the `shred` utility, which provides synchronous file overwriting. The authors of [18] provided a tool called FoSgen, which adds file system extensions in order to overwrite deleted files automatically and immediately. FoSgen *"intercepts file system events that require overwriting and move the corresponding files into a special per-file-system directory called `ForSecureDeletion`"* [18]. The `shred` tool will then, either manually or automatically using a `cron job` overwrite the data in the `ForSecureDeletion` folder.

The second method consists of two patches for the Ext3 file system. The first patch overwrites all files, or files marked with a secure deletion attribute set in the Ext3 and Ext2 file systems, once. The second patch supports multiple overwrites, but also deletes the meta-data (i.e. user ID, access, modification and creation times) and directory entry [18].

5.3 Distributed Hash Tables

The ENISA report [14] mentions Vanish, a concept that uses Distributed Hash Tables to store encryption keys for a specific period of time. In this section we summarise how Vanish and other concepts, that use Distributed Hash Tables, work. We first briefly describe what a Distributed Hash Table (DHT) is.

A Distributed Hash Table (DHT) *'is a distributed, peer-to-peer (P2P) storage network consisting of multiple participating nodes'* [19]. Data that is to be stored in the network consists of an (index, value) pair, where each node manages a part of the entire index. A DHT uses three general operations: look up, get, and store. A client stores data by first looking up which nodes are responsible for the index, and then storing the data to the responsible node. That node saves the (index, value) pair in its database. Process of retrieving data is similar. First the node responsible for a specific index is looked up, after which the client issues a get request, which returns the value corresponding to the index [19].

The property that make DHTs interesting for the purpose of data erasure is the constantly changing structure of the storage network. In a DHT environment, nodes constantly join or leave network (i.e. churn) [19], which *'means that data stored in DHTs will naturally and irreversibly disappear over time as the DHT evolves.'*

5.3.1 Vanish

Similarly to the Ephemerizer concept, the Vanish proof of concept stores encryption keys externally. Where the Ephemerizer relies on specific trusted third parties (the Ephemerizers), Vanish uses DHTs to store the encryption key. Thus, eliminating the centralised model of the Ephemerizer.

Vanish encrypts data D, with a randomly generated key K and obtains ciphertext C. It then splits key K into N pieces, creating $K_1...K_N$ shares of the key, using a process known as Shamir's Secret Sharing[24]. A threshold T is set to determine how many parts of key K are necessary before K can be reconstructed. Data becomes unavailable when more than $N - T$ parts are lost. The shares $K_1...K_N$ are then randomly divided across nodes in the DHT by choosing an access key L. Feeding L into a pseudo random number generator (PRNG) determines the indices $I_1...I_N$. The shares $K_1...K_N$ are stored at indices $I_1...I_N$ respectively. Eventually, the following is stored in the network: (L, C, N, T). To decrypt the data, Vanish first extracts the access key L, and uses it to derive the locations of the shares of key K. It retrieves the number of key shares specified by N, and reconstructs K. After which C is decrypted to obtain the data D [19].

Due to the churn property of DHTs, over time, shares $K_1...K_N$ are lost. Once the threshold is reached, decryption is not possible any longer, and data is forgotten.

[24] Shamir's Secret Sharing makes it possible to reconstruct a secret using some parts, or all, of the secret.

5.3.2 Comet

Comet is a proof of concept similar to Vanish. It also stores data by creating an (index, value) pair, but additionally stores *'handlers'* to create an Active Storage Object (ASO) [20]. The handlers specify how an ASO can behave. The following events can take place on a node, and can be used to create certain ASO behaviour [20]:

- onGet: *'invoked when a get is performed on the ASO;'*
- onPut: *'invoked upon initial put when the object is created;'*
- onUpdate: *'invoked on an ASO when a put overwrites an existing value;'*
- onTimer: *'invoked periodically.'*

For example, the events can be used make ASOs only available for a specific period of time by setting a timestamp for the object. By requesting the system time and comparing it to the timestamp, an ASO can determine if it should delete itself, or not. ASO behaviour can also be defined to delete itself after it has been read. It is even possible to set an ASO to delete replicas from other nodes, or maintain audit trails *'e.g., indicating where it has been stored thus far, who has read or updated the object, etc.'* [20].

5.3.3 Freenet

Freenet uses the decentralised architecture of DHTs to store data instead of encryption keys. Each node in Freenet has a specific storage space, which forms a virtual file system together with the storage space of all the other nodes. At some point in time the storage space of a node can become full. The node then removes the least recently used data. In doing so Freenet decides whether a file is meaningful (i.e. popular), by keeping the most regularly used file, until the file becomes less used over time. In chapter 6 we will explain how Freenet works in detail.

5.4 Other network related storage methods

In this section we briefly describe storage methods, which do not fit into the above mentioned categories of cryptography or DHTs.

5.4.1 EphPub

Ephemeral Publishing (EphPub) is related to the concepts (e.g Vanish) that rely on DHTs to store and delete keys or data. However, EphPub uses the DNS caching mechanism to store and revoke the encryption keys, such that the problems of a Sybil attack against a DHT can be mitigated [21]. The Sybil attack against Vanish for example, can recover keys for 99% of the Vanish messages by crawling the DHT and saving each stored value before it expires. For more details on the Sybil attack on Vanish we refer to [22].

The property of DNS that EphPub relies on, is that a DNS resolver caches the response to a recursive DNS query to serve further requests, and stores for a period of time, i.e. a Time To Live (TTL) value. The record is erased once the TTL has expired.

As with other concepts, EphPub encrypts a message with a key k. Instead of storing the key on a keyserver, it is distributed, per bit, on separate DNS cache resolvers. As the DNS cache expires according to the TTL, the encryption cannot be recovered, i.e. data is erased. It works as follows:

The sender S generates a list of random DNS cache resolvers, a list of random valid domain names with TTL close to the storage period [21]. Then sender S randomly tests if the different domains are already in the cache of a resolver by sending a non-recursive DNS request. If the domains are in the cache they are removed from the list.

Next the key k is distributed. When a bit of the key k is 1, the sender S *'performs a recursive DNS request of an existing domain name,...,to a DNS cache resolver, res.'* The resolver R replies the record *'and caches the response for a period of time corresponding to domain's TTL'* [21]. If a bit of key k is 0, sender S performs a non-recursive DNS request, which will reply a 1 if the domain name is cached, but a 0 if it is not in the cache.

The sender sends the following to the receiver: the ciphertext of the message, the list of DNS cache resolvers, the list of used domain names, and the expiration time. The receiver R checks the current time against the expiration time upon receiving. If the time is still valid, the bits of key k are retrieved, by using non-recursive DNS query for every domain name. In the case that a record is replied, the bit is a 1, and otherwise a 0. Once the key is retrieved, the ciphertext can be decrypted to reveal the message.

5.4.2 Parasitic Data Storage

A completely different approach to store data comes in the form Parasitic Data Storage. This manner of storing data is more volatile than any of the previous options, because is does not rely on storage media. In fact, it stores data by sending it back and forth between a server and reflector node, using Internet Control Message Protocol (ICMP) packets. When data needs to be erased, the server side simply drops the packets.

However, if there is a malfunction with the connection, the stored data will also be forgotten. This makes Parasitic Data Storage (PDS) more volatile than encrypted data of which the key is erased. With the latter, there might still be hope to brute-force the key, with PDS the data is definitely gone.

5.5 Table

Based on the implementations, or concepts described above, we can create the following table:

		Ephemerizer	Vanish	Comet	Freenet	EphPub	Volleystore	FoSgen
Methods	Changing permissions			P				
	Editing credentials			P				
	Modifying search results							
	Removal of pointers			**X**				**X**
	Moving data				**X**			
	Destruction of storage medium						**X**	
	Data is damaged				P		P	
	Data is transformed	**X**	**X**		**X**	**X**		**X**
Factors	Time	**X**	**X**	**X**		**X**	**X**	
	Meaning							
	Regularity			P	**X**			
	Space				**X**			
		Cryptography	DHT			Other network storage		Overwriting

Implementation / concepts

The **X** indicates which of the methods and factors the concept uses. The **P** is used to indicate which of methods and factors could possibly be used by the concept. The table shows three concepts, Comet, Freenet, and Volleystore, that use interesting methods to delete data from a system. We choose two of these concepts and explain them in more detail in next two chapters.

In chapter 6 we elaborate on Freenet. Freenet claims to actually delete data, compared to Vanish which deletes keys. It also uses two factors (regularity and space) that are not used by other concepts, to determine if data should be deleted. Chapter 7 explains how Volleystore works. We choose this concept, because it is the only one that actively deletes its storage medium. It actually drops the ICMP packet which stores a part of the data, and does not rely on physical storage media. We omit Comet, because it only relies on the time factor, and does not properly explain how data is actually removed.

6 Freenet

In this chapter we explain Freenet, a concept that does not store data indefinitely, but relies on certain factors to determine if data should be erased. Freenet was developed in 2000 to combat the growing concerns regarding information privacy, censorship and surveillance. It uses a decentralised architecture to combine the unused disk space of participants to create a virtual file system [23]. As this disk space is limited, Freenet removes irrelevant data if storage space is running out. We describe the architecture and how Freenet works below.

6.1 Architecture

The most recent version of Freenet has two different network modes: Opennet and Darknet. A participant can choose to run either Opennet, or Darknet, but a hybrid option is also possible. In Opennet mode, a node connects to other unknown (i.e. untrusted) nodes[25], while in Darknet mode a node only connects to trusted ones[26].

In this section we briefly describe the initial architecture of Freenet as explained in the master thesis of Ian Clarke [24]. We then focus on the more recent Opennet architecture, because this is how currently most nodes are connected and exchange data[27]. As the name implies, Opennet is publicly accessible, which means that we can place it on the open system side of Figure 2. Therefore, the erasure of information has the biggest effect in the Opennet architecture. We omit describing the Darknet architecture, because it is used by a smaller amount of nodes, and resides more on the closed system side of Figure 2.

6.1.1 Adaptive network

Freenet is based upon a peer-to-peer (P2P) architecture in which all participants run a node. Each node provides connectivity, offers storage space to the network, and stores some data. A node can receive a request for data. If the node does not have this data stored, it forwards the request to another node, which is more likely to contain the data. When the requested data is found, it is returned through the nodes that forwarded it. Along the way all nodes in the chain remember where the information was stored, and also store the requested data themselves. The latter serves as a cache for future requests, which do not have to be forwarded, but can immediately be replied to with the data [24]. The caching feature allows for data to move through the network, to places where it is in demand. For example, data stored in the Freenet network at point A, and in high demand at point B.Through the requests coming from point B, the data from point A will move closer to point B. Nodes near point B will have a copy of the data, and will serve additional requests coming from that area. Nodes at point A may not receive any more requests, which means that the data takes up valuable storage space and it is better to remove it.

[25] https://wiki.freenetproject.org/Opennet, accessed on 26-06-2015
[26] https://wiki.freenetproject.org/Darknet, accessed on 08-06-2015
[27] https://wiki.freenetproject.org/Opennet, accessed on 26-06-2015

6.1.2 Opennet architecture

Freenet's architecture and routing, for both Darknet and Opennet modes, is based on a small-world network topology. This provides Freenet with the following properties[28]:

- *'The network is fully connected, in the sense that there exists a route between any two nodes on the network.'*
- *'There exist short routes from any node to any other node.'*
- *'If A is connected to B, and B is connected to C, then there is an increased probability of A being connected to C.'*

To model its network Freenet uses Kleinbergs small world model with greedy routing [25]. The network can be viewed as a grid, where each node is connected to all of its neighbours that lie within a certain distance[29,30]. To improve on routing, additional long-range connections are added.

As mentioned Freenet uses greedy routing, which means that a node A can choose the next node B in the route, if node A thinks that node B is closer to the eventual destination. To establish a successful route, it is important for nodes to have a location in the network (i.e. grid).

In Opennet mode, a node automatically connects to other nodes in the Freenet network. A new node starts without having any knowledge about the rest of the network, and it chooses its location randomly. To become part of the network, a bootstrap process is used which relies on seed nodes and announcements. A node comes preconfigured with a list of seed nodes[31].

The node connects with a seed node, in turn the seed node announces the node to the part of the network that has a similar location as the node. Nodes with similar locations will reply to the node and it starts operating as normal [25]. The node's location remains fixed and using path folding[32] the small world network toppology is maintained.

6.2 Inserting and retrieving data

Freenet is similar to a Distributed Hash Table, and uses a key mechanism (explained below) to insert data in, and retrieve data from, the network. Meaning, every data item is stored (i.e. inserted) together with a corresponding key, which is needed to retrieve the data from the network. The mechanism is also used to authenticate the data when it is transferred [25].

In this section we describe how data is inserted and retrieved from the network. We first explain the concept of key closeness and how it is related to locations. Then we explain how a route, to insert or retrieve data, is established, and which messages are used.

[28] https://wiki.freenetproject.org/Small_world_topology, accessed on 29-06-2015
[29] Also known as lattice distance, or Manhattan distance
[30] https://wiki.freenetproject.org/Kleinberg_network, accessed on 29-06-2015
[31] https://wiki.freenetproject.org/Opennet, accessed on 27-06-2015
[32] https://wiki.freenetproject.org/Opennet, accessed on 29-06-2015

6.2.1 Key closeness and locations

Freenet relies on a concept called key closeness, which is described using the following axioms [24]:

- *If A is closer to B than D is, and E is closer to B than A is, then E is closer to B than D is*
  Which is formally represented as: $A <_B D \wedge E <_B A \Rightarrow E <_B D$
- *If A equals B and D does not equal B, then A is closer to B than D is*
  Which is formally represented as: $A = B \wedge D \neq B \Rightarrow A <_B D$

The letters (A,B,D, and E) resemble the keys. The axioms allow a key to be compared with a list of keys, resulting in the closest key from the list. The key closeness concept is used to build routes using the node's routing table.

As already mentioned, a node running Opennet mode chooses its location when it is created and remains fixed. To insert and retrieve data from the network, a key that corresponds to the data is used. This key is also provided with a location, which ensures that data can be routed to a node with a similar location. In other words, the key and corresponding data are routed to a node based on the distance (i.e. key closeness) between the location of the node and the location of the key. The locations given to keys and nodes are in the form of a number (r): $0 \leq r < 1$. We describe in section 6.3 what the location of the key is based on.

The location related to a key is also known as a routing key. As already mentioned, data requests and inserts are *'routed to the node whose location is closest to the data location'*[33]. Routing relies more on the network topology than on the actual routing algorithm. The algorithm requires that there is a balance between, *'connections to nodes with nearby locations, and connections to nodes with distant locations'*[34].

6.2.2 Routing to insert or retrieve data

A route needs to be established to insert and retrieve data, consisting of an ordering of neighbouring nodes that gradually reside closer to the key [25].

For example, to find data that matches a certain key, a route for that key is built. First a node A determines which of its neighbours is closer (based on its location from the routing table) to the requested key. Node A forwards the request to the closest node B. Node B checks if the key exists in its database. If it does, the data is returned to the node that made the request; in this case node A. If node B does not have the key, it forwards the request to the node which location is again closer to the location of the key. In the situation where it is not possible to determine if a node B's neighbours are closer to the key, the request is backtracked to the previous node (Node A) where the next best node (Node C) is chosen. This process continues until the request reaches its

[33] https://wiki.freenetproject.org/Routing, accessed on 08-06-2015
[34] https://wiki.freenetproject.org/Routing, accessed on 08-06-2015

goal, or the maximum number of steps (Time To Live) that were allowed have been reached. When the requested data is found, the route is aborted and the data returned to the previous node. The returning of the data continues back through the route until it reaches the originating node, where every node caches a copy of the data. In the recent version of Freenet (v0.7) it is possible for a node to search the cache of its neighbours [25].

A similar process occurs when data is inserted (i.e. stored) in the network. When the route terminates, without the data matching the key being found, the information is send up the route. In the case where information is found, it is returned (as discussed above), which means that a data insert with the same key fails. As already mentioned, in both cases (inserting and retrieving), nodes along the route store the information in their cache, such that future requests can be served immediately.

6.2.3 Messages

Freenet uses four types of messages to insert and retrieve information from the network: Data Request, Data Reply, Request Failed, and Data Insert. Each message contains the following parts [24]:

- ID: An ID uniquely corresponding to its operation
- TTL: A Time To Live (TTL) value, which is decremented when a node passes the message on. The TTL *'indicates how many times the massage should be passed on'*, and if the value equals zero, the node should return a Request Failed message to the sender.
- Source: Who sent the message (not who initiated it). To clarify, an initiator (e.g. Node A) of a request determines the ID and the initial TTL. It then sends the message to a different node (e.g. Node B), who in turn forwards it again. That node (i.e Node B) becomes the sender, the identity of the initiating node (i.e Node A) is not stored in the message.
- Destination: Who the recipient of the message is.

A Data Request message requests for data that corresponds to a specific key. If a node has the matching data, it replies with a Data Reply message containing the data. If the data is not present on a node it should forward the Data Request message to a node which is more likely to have the data. The ID and source of the message is to be stored by a node, such that Data Reply, or Request Failed messages, can be handled and returned along the route.

A node sends a Data Reply message (as a reply to a Data Request) if the requested information is found, or if it receives a Reply it previously forwarded a Request for. The node retains a copy of the replied data, and stores a reference to the node that the Reply was sent to.

If the data is not found, a Request Failed message is sent by the node. Similarly to the Data Reply, a node should only receive a Failed message it previously forwarded the Data Request for. If the TTL of a message becomes zero and the data is not found at the node, a Request Failed message has to be sent back to the node which forwarded the Data Request previously. If a Failed message is

received and the TTL is not yet zero, the node should forward the Data Request to the next node until all its neighbouring nodes have been explored. Then, the Failed message is sent to the node that made the Request previously.

To insert data into the network a Data Insert message is used, which consists of the key corresponding to the data and the data itself. A node caches the key and the data, then forwards the Insert message to the next node which has a location closest to the location of the key. This process continues until the TTL reaches zero and makes sure that data is spread out over several nodes with locations close to key location. The latter increases availability of data, because nodes go offline at random. To prevent loops a node should send a Request Failed message if it sees the same Insert message as before.

6.3 Key mechanism

We discussed above that a user sends an insert, or request, message to store a file in the network. The insert message consists of the file and a key. In the paper [26] that followed the master thesis, more details were provided on the key mechanism used by Freenet. We checked this against a more recent description of the types of keys used, on the Freenet wiki page[35], and a paper describing an improved architecture of Freenet [25]. We explain the key mechanism.

Freenet uses two methods to generate keys: cryptographic hashing of a file, and a bitwise XOR of a hash of a document name and a fingerprint of a public key (from an asymmetric key pair) which signs the document [25].

The two methods create two basic key types used by Freenet: a Content Hash Key (CHK) and Signed Subspace Key (SSK). Two other keys are build upon the basic keys: Updateable Subspace Key (USK) and Keyword Signed Key (KSK).

A CHK consists of three parts: a hash of the file, the decryption key, and settings for used algorithms. It is used for referring to large static pieces of data [25]. Hashing the file creates a unique file key of that file[36]. The file is encrypted with a randomly generated encryption key. The decryption key is published together with the hash of the file, not with the file itself. The reason for this, is to allow everyone to decrypt and use the file, while providing plausible deniability for the person running a node. It is important to note that plausible deniability in this sense is trivial, because the key is publicly available. Therefore, the claim that it is impossible to know the plaintext due to encryption is unlikely to hold (e.g. in court).

A SSK is used to create a namespace which can be read by anyone, but only written to by its owner. The SSK relies on an asymmetric key pair, such that the person with the private key can update to the namespace. It is used for documents that require regular updating, such as websites. Also the SSK consists of three parts: a hash of the public key, the document decryption key and

[35] https://wiki.freenetproject.org/Freenet_key, accessed on 03-06-2015

[36] According to the paper [26] SHA-1 is used to generate the hashes. However, according to the more recent Freenet wiki page (page was edited in July 2014), SHA-256 is used as the hashing algorithm. For this thesis, we assume that SHA-256 is used.

the encryption settings. Only the first part, the hash of the public key, differs from the CHK, and is used to uniquely identify the document.

In order to find documents in Freenet, the public key of publisher needs to be found, which is made possible by expressing the keys (CHK and SSK) as a Uniform Resource Identifier (URI) [25].

We also mentioned that routing relies on the location of a node and the location of the key. A CHK receives a location based on the hash of its data, and the location of a SSK is based on the hash of the document name and public key.

6.4 Data storage

Freenet uses a stack mechanism (explained below) to store keys and corresponding data, which allows a Freenet node to remove information based on its popularity [24]. The node stores information that has been inserted in the network via that node, or information that passes through the node following a request. If the storage space is full, *'the least-recently-accessed piece of information is deleted'* [24].

The thesis [24] points out that some forms of popular information (e.g. pornography) can be considered, by Freenet, to be more valuable than less popular information (e.g. an academic paper). However, information that is less popular, but important to some people, will still be requested by a small number of nodes. Thus, the information will be stored close to the requesting nodes, but will be less dispersed throughout the entire network.

It is indicated in the thesis [24] that Freenet serves the need of the public as a whole, and information which is relevant to only a minority of the Freenet users, will eventually be removed from the system. Additionally, as storage capacity increases faster than the information output, Freenet might only remove information that is never accessed [24].

The paper discussing the addition of the darknet option in Freenet [25], mentions that a node has two types of storage: a short term cache and a long term datastore. The short term cache deals with, as mentioned, immediate requests, and contains data that passes trough the node i.e. as part of a Data Reply. This cache fills up quickly, and when its full the oldest entries are dropped, which would mean that data could not be stored for long[37].

The data store is for data that needs to be stored for longer periods. Data is stored in the long term cache when the location of the data's key is better placed at the node than the locations of the node's neighbours, and only data that is inserted, and not retrieved[38]. For example, a node with location 0.49, will store data with a corresponding key that has a key location of 0.50. Provided the neighbours have locations that are further away [25]. The data store takes longer to fill up, and therefore data can be stored for longer periods in Freenet.

[37] https://wiki.freenetproject.org/Datastore, accessed on 29-06-2015
[38] https://wiki.freenetproject.org/Datastore, accessed on 29-06-2015

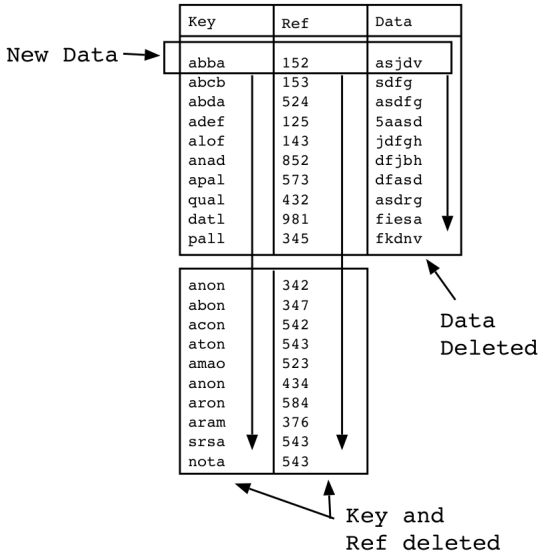

Figure 4: Data storage stack within a node [24]

6.4.1 Stack and data removal

We describe the stack mechanism in more detail. The stack consists of two parts [24]:

> *The upper section stores keys, a reference to the node from the key/ data originated, and the data itself. In the lower section only the keys and the references are retained.*

This means that in the cases that data is inserted, or data is replied, a key, reference and the data is pushed on top of the stack. Data residing at the bottom of the upper section is deleted when it moves to the lower section; the key and reference to the source remain. The effect is that the node can request new copies of the data from the original data source. It is important to note that [26]:

> *That is, there is no "permanent" copy which is being replicated in a cache. Once all the nodes have decided, collectively speaking, to drop a particular file, it will no longer be available to the network.*

6.4.2 Discussion regarding storage time and storage method

To the best of our knowledge there are no publications that have measured the storage time of data, i.e. how long data is retained on Freenet before being deleted. The performance of Freenet has been measured, but focusses on simulations [23, 26, 25] and not on the performance of the actual network.

The most recent paper by Roos et al. [27] does use real network data and discusses file popularity, user activity and content, but only mentions that *'there are*

few very popular files, and the majority of the files is not requested frequently.' Thus not providing an indication on data storage time.

During a conversation with a developer on Freenet's IRC channel[39], it was explained that Freenet currently uses a different method to delete data. This method relies on overwriting and healing. First a file is divided into parts and spread through the network. At least 50% of the parts are necessary to successfully retrieve the file. Overwriting deletes the parts, while healing recreates parts that were retrieve successfully. This has a similar effect as the stack mechanism, but with the advantage that files with less than 50% of the parts will not get healed. If the parts do not get healed, they eventually be overwritten and thus erased. Unfortunately, we were not able to verify this method in scientific papers or on the Freenet wiki page.

However, Freenet does have a page which provides statistics including the data storage time (i.e. data lifetime); see Figure 5. We can observe from Figure 5 that data has a less than 25% change to be retrieved successfully. It is important to note that this is based on the new deletion method, which we were not able to verify and that in order to access the statics page Freenet needs to be installed[40].

6.5 The Right to be Forgotten and Freenet

Our above mentioned analysis of Freenet shows it deletes data after a certain period of time, and we can infer several factors of forgetting that contribute to this.

Freenet relies on the following factors: storage space and popularity of data. A node will only consider to remove data when its assigned storage space (i.e. the stack) is full. An entry (i.e. key and corresponding data) that resides at the bottom of the stack if it has not been used recently. When a new entry is pushed on top of the stack, the less recently used entry falls of. The space it used becomes available for fresh entries.

Thus, data that is less frequently requested will move to the bottom of the stack, and fall off if the storage space is full and no new requests for the data enter the node.

If we regard deletion in Freenet with respect to the factors of forgetting mentioned in chapter 3, we can state the following:

> Unpopular data in Freenet can be considered as meaningless information with respect to all Freenet participants. The meaninglessness of data is measured according to regularity of use over time, where the time period is determined by the speed with which the storage space runs out. The lack of storage space can be considered as the factor which actively tells the system that erasure of data should take place.

[39] # freenet on chat.freenode.net, see https://freenetproject.org/irc.html

[40] The graph can be viewed at:
http://127.0.0.1:8888/freenet:USK@sCzfyxndzgtfzoPRUQExz4Y6wNmLDzOpW04umOoIBAk,~X5tIffdcfNWPIKZ2tHgSGPqIk9au31oJ301qB8kTSw,AQACAAE/fetchpull/1534/

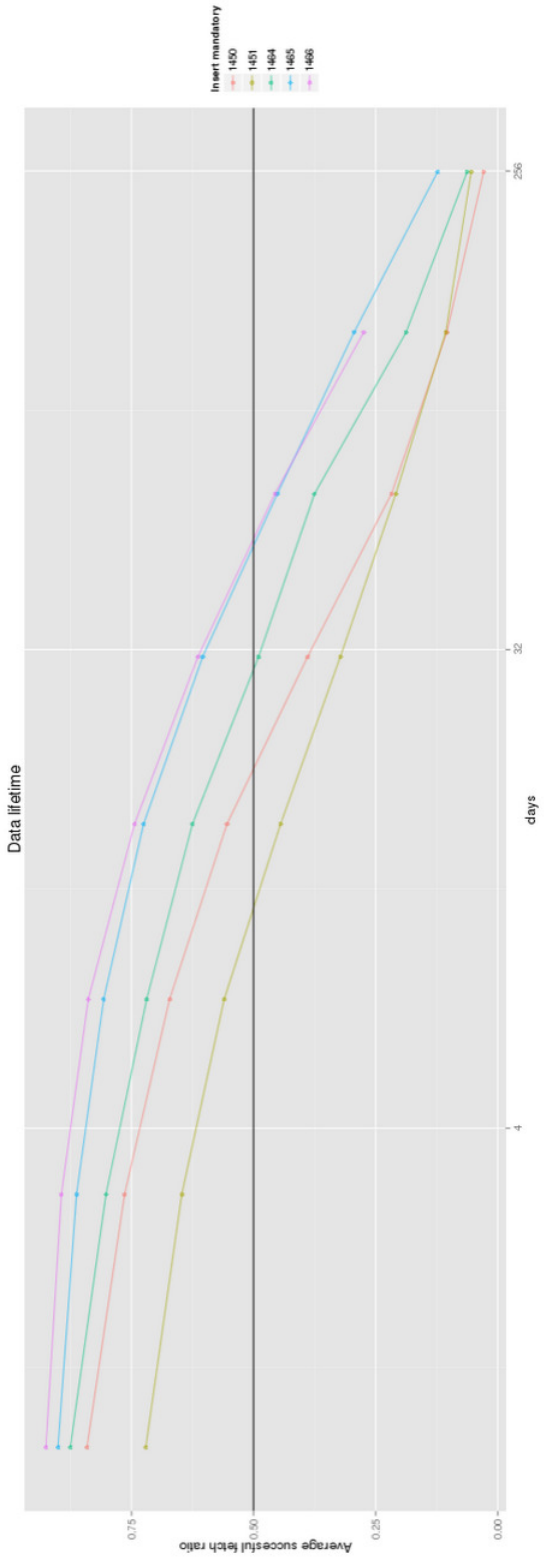

Figure 5: Graph of data lifetime of several recent versions of Freenet.

It should be noted that deletion of data is as unpredictable as its availability, because nodes can appear and disappear at random. The reason for this, is the decentralised architecture (P2P) of Freenet. Every node is considered equal and can decide to go offline at any time either to, come back online later or never.

If a node with its stored data goes offline for a long period and the network forgets about it (i.e. the routing tables do not have any entries to the node), data is considered erased. When the node comes back online again and advertises itself, data can become available.

7 Parasitic Data Storage

Parasitic data storage (PDS) is a different form of volatile storage. It uses a network (i.e. the Internet) to continuously send data between nodes, utilising the time it takes for data to return to the node, as a way to briefly store data in network [28, 29]. Thus, PDS avoids the use of physical storage devices (e.g. hard disk drives).

As the name suggests, PDS only benefits one party and uses resources at the expense of another. Therefore, the concept has been considered a form of attack. The party using PDS does have the advantage that data is not stored on their storage media [29], which might be beneficial in certain situations. For example, if sensitive data stored on a hard disk might pose a threat to a person, PDS allows one to avoid physical storage, and offer a degree of deniability [29].

Another benefit is PDS's method of data erasure compared to erasure on physical storage media. Completely erasing data from a storage medium is difficult, because data can still be recovered after it has been overwritten, or marked to be reused by the operating system [18]. PDS avoids physical storage devices, erasure of data relies on the time factor, and the probability that a piece of data gets lost in transit.

7.1 First concept

The first concept of parasitic data storage that used the Internet was established by Wojciech Purczynski and Michal Zalewski in a paper [28] published in Bugtraq (an email list).

As we mentioned the idea behind PDS is that data is not stored on a physical device, for example a server, but in the "cables" connecting two or more nodes. For example, when data is send from Node A to Node B it is first split up into packets. When a packet is sent to Node B, it is not stored on Node A any longer, and neither on Node B; it is stored in the route towards Node B. By using a protocol that replies to the data packet with the same content, a cycle can be built which can be used store and retrieve data.

The paper [28] distinguishes between two classes of storage: memory buffers and disk queues. The memory buffer class (class A), relies on storing data while it is in transit. It is likely that the packets will be cached and processed in memory and not moved to a physical disk. Class A has the following properties [28]:

- Low latency (miliseconds to minutes), making it more useful for near random access memory applications,
- Lower per-system capacity (usually kilobytes), making it less suitable for massive storage,
- Only one chance to receive, or few retransmits, making it less reliable in case of a network failure,
- Data not likely to be stored on a non-volatile medium or swapped out, increasing privacy and deniability.

However, once a network connection is lost, so is the data on that channel. Class A storage should therefore not be used for critical data that needs to be preserved.

Class B storage uses disk queues to store data, for example, by bouncing mail between a client and server. This class of storage abuses the property of a Mail Transfer Agent to store emails for some time, if the email cannot be delivered immediately. For example, an (ab)user would have to determine the delivery timeout per server; basically determine for what period of time the server will try to deliver the email. The data will then be send to one or, if data is split up, multiple servers, while the incoming port[41] is blocked. Effectively preventing email being delivered by the server. If the port remains blocked, the server will eventually stop and delete the email from the queue. Therefore, in order keep the storage process going the incoming port needs to be unblocked before delivery timeout runs out; such that the mail can be delivered, resend, and the port blocked again. When there are multiple servers involved, a specific server needs to unblocked, such that not all servers deliver there email [28].

To retrieve the stored data, the (ab)user would have to keep track which servers are storing the data and unblock the ports of those servers.

Class B storage has the following properties [28]:

- High per-system capacity (megabytes), making it a perfect solution for storing large files and so on,
- Higher access latency (minutes to hours), likening it to a tape device, not RAM (with exceptions of SMTP hosts that accept ETRN command to immediately re-attempt delivery),
- Very long lifetime, increasing per-user capacity and reliability,
- Plenty of delivery attempts, making it easy to recover the data even after temporary network or hardware problems,
- Likely to leave a trace on the storage devices, making it a less useful solution for fully deniable storage (although it would still require examining a number of foreign systems, which does not have to be feasible).

The paper [28] suggested the storage capacity can be calculated as follows:

$C_{max} = L * T_{max}/P_{size} * D_{size}$

Where,

- C_{max} is the maximum capacity in bytes;
- L is the maximum available bandwidth ($bytes/second$);
- T_{max} is the maximum lifetime of a single packet, which includes packet queuing and processing delays;
- D_{size} is the *'size of a packet required to store an initial portion of data on remote host'*;

[41] The port that the mail will be delivered to.

- P_{size} is the *'size of a packet required to sustain the information stored on remote host'.*

The precise meaning of D_{size} and P_{size} remains unclear in the paper [28]. However, we assume that D_{size} is used to initialise the storage of data and that P_{size} consists of some error correction scheme, which sustains the information. The paper presents the following indication of storage capacity:

Bandwidth	Class A	Class B
28.8 kbps	105 MB	2 GB
256 kbps	936 MB	18 GB
2 Mbps	7.3 GB	147 GB
100 Mbps	365 GB	7 TB

7.2 Volleystore

The first concept of PSD, mentioned above, was further discussed in a paper [29], which also demonstrated a proof of concept.

The paper states that there is a trade-off between storage capacity and access latency. Using low-delay channels results in low-latency access, but data packets need to be injected more often into the network, which takes up bandwidth. On the other hand, high-delay channels allow for more storage, but also high-delay access latency [29]. In order to provide guarantees regarding availability, integrity, and confidentiality of the data, the following goals were stated [29]:

- Storage capacity: *'Maximum amount of data that can be stored in the system with very low probability of data loss.'* This depends on how error prone the protocol used is. For example, the drop rate of Internet Control Message Protocol[42] (ICMP) packets influences the storage capacity.
- Availability: *'Maximum latency of the system to reconstruct the stored data.'* The availability can be improved by inserting copies of packets or using *'denser error correction codes'.*
- Confidentiality: *'Percentage of information revealed if storage nodes are compromised.'* To ensure confidentiality, encryption should be used. Inserting multiple copies would increase availability, but also hide the actual amount of data stored.

A basic PDS system consists of a server node, a reflector node, and a path which connects the server and the reflector [29]. Taking the ICMP packets as the storage vessel, the paper states the following parameters, which determine storage capacity and access latency [29]:

- *'payload of the messages,'*
- *'error rate on the path that connects the server and reflector nodes,'*
- *'the speeds at which the two nodes can input and output data'*

[42]The Internet Control Message Protocol is used for diagnostic purposes in a network, e.g. to check if a server can be reached

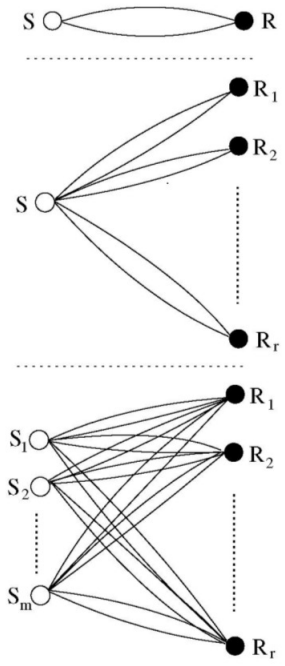

Figure 6: Three settings for parasitic storage system [29].

- *'the round-trip time between the two nodes.'*

The paper also describes three possible setups of a PDS system which uses ICMP packets; see Figure 6.

In a single server and reflector setup, the ICMP packets are sent between two nodes, which is also called a storage channel. It requires multiple packets to be transmitted such that data-availability can be maintained. The setup is formally defined as follows [29]:

$(\gamma_S, rtt, \gamma_R)$ where,

- γ_S: the input/ouput throughput rate of the server node, in packets per second.
- γ_R: the input/ouput throughput rate of the reflector node, in packets per second.
- rtt: the round-trip delay time between the nodes

The information capacity of the channel *'determines the maximum average number of bits that can be sent per channel use with arbitrarily low probability of error'*, and can be described as [29]:

$C = (1 - P_e) \times nbits/symbol$

Where P_e stands for the probability with which each transmitted symbol is erased. The meaning of 'symbol' is not clearly explained in the paper [29], we assume it applies to the packets being send on the channel. In other words, P_e

stands for the probability that a packet lost while being stored on the channel. The number of packets that can be stored depends on minimum value of both throughput rates.

The information capacity is in the following formula to describe the storage capacity of this channel using this setup:

$C_s = rtt \times min(\gamma_S, \gamma_R) \times (1 - P_e) \times n$ where,

- $min(\gamma_S, \gamma_R)$, provides the minimum throughput of both nodes
- $(1 - P_e)$, the probability that a packet is not lost.
- n, the number of bits send on the channel

It is important to note that the route between γ_S and γ_R is not a direct one. A packet passes through several devices, not part of the Volleystore scheme, to get to γ_R. These devices have different throughputs, which might increase the delay or eventually drop a packet, and are part of the formula as $1 - P_e$. For example, the value of P_e is increased if a device in the route drops packets due to a high load, which means that the probability that a packet is not lost, goes down. Thus the capacity C_s decreases.

The maximum access time of the storage channel is $2 \times rtt$. One rtt, because that is the maximum time at which a packet is considered lost. The other rtt comes from the server waiting for a packet that is part of the error correction scheme, which also might be lost. Volleystore uses two error correction schemes: parity and replication. Both are explained below, and rely on a form of redundancy, which adds to the total amount of packets that need to be send on the channel.

The setup of a single server and multiple reflectors has two issues regarding storage capacities which depends on the throughput[29]. If the throughput of the server node channel is larger than the combined throughput of all the reflector nodes, the maximum storage capacity can be reached. This is defined as follows:

$C_s = \sum_{j=1}^{r} C_{s_j}$ where,

- Capacity per channel is $C_{s_j} = rrt_j \times \gamma_{R_j} \times (1 - P_e j) \times n$
- Amount of reflector nodes is: r

When the throughput of the server node channel is smaller than the combined throughput of all the reflector nodes, the best channels have to be picked in order to maximise the storage capacity. It is defined in [29] as:

$C_s = \sum_{j=1}^{r} C^*_{s_j}$ where,

$C^*_{s_j} = \gamma^*_j \times rtt_j \times (1 - P_e) \times n$, such that

$\gamma_S = \sum_{j=1}^{r} \gamma^*_j$.

Where γ_S defines the maximum storage capacity and the access latency depends on the highest rtt of all used channels.

The corresponding γ^* values that S can utilise can be chosen using the following formula [29]:

$$\gamma^*_{i_j} = \begin{cases} \gamma_{R_{i_j}}, \text{ if } 1 \leq j \leq t, \text{ s.t.} \\ \sum_{l=1}^{t+1} \gamma_{R_{i_l}} > \gamma_S \geq \sum_{l=1}^{t} \gamma_{R_{i_l}} \\ \gamma_S - \sum_{l=1}^{t} \gamma_{R_{i_l}}, \text{ if } j = t+1, \\ 0, \text{ if } t+2 \leq j \leq m \end{cases}$$

The last setup consist of multiple server and connector nodes; see Figure 6. There are an m-amount of servers and an r-amount of reflectors, and only the available throughput is the limiting factor. Meaning that among all available channels, the best channels have to be chosen.

To achieve maximum storage capacity, the best channel has to be chosen, and its capacity has to be, iteratively, added up with the next best channel. The process continues until all available throughput for servers and reflectors has been used.

The storage capacity of one channel is defined as:

$C_{S_{i,j}} = rtt_{i,j} \times min(\gamma_{S_i}, \gamma_{R_i}) \times (1 - P_{e_{i,j}}) \times n$ where,

- The amount of servers $1 \leq i \leq m$
- The amount of reflectors $1 \leq j \leq r$

The overall storage capacity is defined as:

$$C_s = \sum_{i',j'} C_{s_{(i,j)}}$$

The latency remains the same as with the previous set-up.

The implementation of Volleystore consist of two programs: volleyctl (client) and volleybot (server). Two types of blocks are used: storage blocks and volley blocks. User data is split into parts and put in storage blocks. A volley block is obtained by expanding a storage block with Error Correcting Code (ECC) or replication [29]. The volley blocks are the actual packets that are sent to the reflector node.

The Volleyctl uses the following methods [29]:

- Inject data, which divides user data into 1KB storage blocks and packs them into ICMP packets. The ICMP packets are sent to Volleybot with some metadata such as the block number.
- Retrieve data, requests Volleybot to retrieve a storage block, receives the data from Volleybot and delivers it to the user. We assume that Volleybot uses the metadata (i.e. block number) to retrieve the necessary volley blocks to rebuild the user data, the paper [29] is not clear on this.

- Delete data, requests Volleybot to delete a block with specific block number. Meaning that it erases its metadata, resulting in the packets being dropped.
- Get statistics and timecheck, are used for diagnostic purposes for the health of the system.

The Volleybot server maintains the packets that stored on the wires, which includes inspecting the returning (reply) packets. When a reply arrives, the payload is *'verified against a locally stored checksum, packed into the payload of a new ICMP echo request packet, and sent back to the reflector'* [29]. To maintain availability, Volleybot uses two error recovery schemes: parity and replication.

With parity, each storage block is split up in two volley blocks. A third volley block is created and contains the byte-wise XOR of the previous two volley blocks. Any two of these blocks can replicate the third when needed, which is then inserted back into the storage channel. With replication, the volleyblocks are copied and injected into the channel. When a packet is missing, Volleybot waits until a copy arrives.

7.3 The Right to be Forgotten and Volleystore

The advantage of PDS over the other concepts, is that it deletes its storage media, i.e. the ICMP packet. Other concepts, such as encryption, do not. Therefore, there is always a possibility that data, which was thought to be erased, might reappear. For example, when encryption is broken or storage server comes back online.

A similar solution would to automatically destroy storage media, but this is infeasible. For example, economically as the costs of replacing destroyed media would be high, and error prone as other data stored on the same medium might be destroyed too.

The disadvantage is that PDS is highly volatile, e.g. as soon a connection is lost, or a server goes down for even a couple of minutes, the data is lost forever. This makes PDS not the ideal option for critical data applications, such as health or financial records.

8 Discussion

In this thesis we have explored the Right to be Forgotten from legal, social, and technological perspectives. We have also further investigated two concepts that use the methods and factors of erasure we mentioned earlier in this thesis. In this chapter we discuss the influence of business models that utilise personal data, the efficacy of erasure methods, the feasibility of concepts, and the relevance of data over time.

8.1 Business models that utilise personal data

We mentioned in the introduction, see chapter 1, that there are companies whose business model depend on collecting and storing as much personal data as possible. The market (e.g. personal advertising) they use the data for is highly lucrative. Due to the interconnectivity of the Internet, it is possible for those companies to have customers, i.e. users or data subjects, across the world. This makes enforcing a right like the Right to be Forgotten difficult, because the legislation of the country in which the company resides does not match with the legislation of the customer's country. It also depends on the legislation having an actual impact when it is enforced. For example, at the moment fines can be low enough[43], such that paying the fine and continuing to violate the law is still more lucrative than to conduct business legally. If the RTBF cannot be enforced properly, companies might choose not comply with it and continue to store personal data indefinitely.

It remains to be seen if the upcoming General Data Protection Regulation (GDPR) can have that impact. Although it looks promising. For example, non-European companies have to comply with European rules, and fines can be imposed upto 2% of the annual turnover. This could lead to business models that depend on collecting, and storing data for a specified amount of time.

8.2 Efficacy of erasure methods

We explained in chapter 3 that erasure is not equal over all domains. It is therefore difficult to predict how the effects of a measure to delete personal information will be accepted by people. This depends on the amount of people that are affected by the measure.

An example that illustrates this, is the erasure of search results from Google. This erases personal information from the point of view of all Google users[44] that used the search engine as an interface to access the information. However, users of other search engines (e.g. Yahoo, or Bing) might still be able to find the information. From their point of view the data is not erased. The impact of the erasure of search results then depends on the amount of users of a search engine. The larger the amount of users the more people 'forget' the information, unless they start using other search engines.

[43] https://bgr.com/2014/01/08/google-privacy-law-violation-france-fine/, accessed on 08-07-2015

[44] Someone who uses the Google search engine

Another example, would be the concept of deleting data by encryption. If encrypted data, that was thought to be erased, suddenly becomes available again, e.g. due the leak of an encryption key, would people still accept it as a valid method to erase data? This too depends on the number of individuals that are harmed. If only a few are harmed, it might still be considered a valid method. What if personal data of hundreds of people is leaked, when it should have been erased?

8.2.1 An example of a side effect

Another point to consider is that while some measures will prevent personal information from being accessed, they might give rise to services designed to circumvent these measures.

An example of this is the removal of search results. The risk lies in the fact that the 'removed' information can still be found through the search engine by not using specific key words, i.e. a name. The method might increase the effort to find the information for the common man, but could reduce the effect of the Right to be Forgotten if there is a service that finds the information anyway.

This might be achieved as follows: First a keyword is used to create an initial set of results (i.e. tax fraud), these results include everybody somehow related to the keyword. By searching for a name and the same keyword, a smaller set is created. This set does not contain the Right to be Forgotten removed results. By subtracting the smaller set from the larger set, a list of results that include the removed records is left. Thus relevant information found.

However, a relatively specific generic term (e.g. a particular crime) is needed, meaning that there has to be some level of suspicion about the person having done something that could produce a result.

8.3 Feasibility of concepts

Even though we presented several concepts that automatically erase information based on a certain factor, none of them can totally prevent information from being copied in one way or another.

Consider a system that uses encryption to keep information confidential, and to automatically delete data. The system still has to, at some point, decrypt the information, such that a person can use it. In turn the person can copy the information in several ways, e.g. by hand, a print screen, or photo, and redistribute it.

Which means that even if an organisation complies with legislation and implements some measure of erasure, it will never be foolproof.

8.4 Data relevance

It is difficult to determine if information is relevant enough to keep stored, or irrelevant enough for there to be a need to erase it. For example, information

existing in the present could be considered irrelevant to a particular group (A), and relevant to another group (B). However, over time the information could gain in relevance for group (B), making it important information in historical context, for example. This means that it is difficult to decide in the present, if data or information will become important in the future. Which makes it difficult to decide on what criteria a system should use to automatically determine if data should be deleted, or not.

Related to this problem are the topics discussed in chapter 3. The two responses, cognitive adjustment and perfect contextualisation, are difficult to achieve. The amount of data required to achieve perfect contextualisation, is more and would grow faster than the current amount of personal data being stored. For example, in order to determine the context in which a decision has been made, a description of 'why' the decision was made is necessary. This 'why' should then also include the emotional context to fully understand the context in which the decision was made.

9 Conclusion

At the start of this thesis we stated several research questions, which we went on to explore in the rest of the thesis in order to answer the main research question:

> *In what way can the Right to be Forgotten be implemented by technology, such that it fits our societal expectations about forgetting information more naturally?*

In chapter 2 we explained the Right to Erasure, its history, and terminology. Using two examples of case law we have seen the different ways in which data can be considered forgotten in information systems: the erasure of search results and not displaying data. Using that information we created an initial model which indicates where methods of erasure could be applied. However, both methods do not explicitly erase information and we were left with the question, what does it mean for data to be erased?

We answered that question in chapter 3. We took a closer look at what it means to forget data in the human brain and how this relates to an external transactive memory. We refined the external transactive memory concept and applied it to information systems, where we discovered that the erasure of information is not equal across the involved domains. We also explained that an information system needs to be actively told to erase information, and that this can be arranged by certain factors: time, meaning, regularity, and space. However, it did not clarify *how* erasure should take place.

We attempted to model this in chapter 4, by stating that information can be considered erased, if a requesting party fails to access the data. This failure to access can be achieved by: denying access to, and withholding the location of data, or rendering the data uninterpretable.

In chapter 5 we elaborated on several technical concepts and classified them based on the methods of the previous chapter. Using the classification we chose Freenet, a storage network that deletes data based on its popularity, and Volleystore, a concept of parasitic data storage which deletes its storage media. We explained both Freenet and Volleystore in detail, in chapters 6 and 7 respectively.

In conclusion, technology can improve on the current position of remembering-by-default by implementing erasure methods, which are automatically triggered by factors that determine if data should be erased. This would change the current position to one where forgetting-by-default is considered more natural in technology, and thus fits more into our expectations about forgetting. The result would be that Data Subjects would have to rely less on the Right to be Forgotten to have their personal data erased when need be, but instead rely on technology to do this for them automatically.

References

[1] Martin Hilbert and Priscila López. "The World's Technological Capacity to Store, Communicate, and Compute Information". In: *Science* 332.6025 (2011), pp. 60–65. DOI: 10.1126/science.1200970. URL: http://www.sciencemag.org/content/332/6025/60.full.pdf.

[2] Viktor Mayer-Schönberger. *Delete: the virtue of forgetting in the digital age*. Princeton University Press, 2011. ISBN: 978-0-691-15036-9.

[3] World Economic Forum. "Rethinking Personal Data: Strengthening Trust". In: 2012. URL: http://www.weforum.org/reports/rethinking-personal-data-strengthening-trust.

[4] World Economic Forum. "Rethinking Personal Data: A New Lens for Strengthening Trust". In: 2014. URL: http://www.weforum.org/reports/rethinking-personal-data-new-lens-strengthening-trust.

[5] K Schwab et al. "Personal data: The emergence of a new asset class". In: 2011. URL: http://www.weforum.org/reports/personal-data-emergence-new-asset-class.

[6] Paulan Korenhof et al. *Timing the Right to Be Forgotten: A study into "time" as a factor in deciding about retention or erasure of data*. 2014. URL: http://papers.ssrn.com/sol3/papers.cfm?abstract_id=2436436.

[7] European Commision. *Proposal for a regulation of the European Parliament and of the council on the protection of individuals with regard to the processing of personal data and on the free movement of such data*. 2012. URL: http://eur-lex.europa.eu/legal-content/EN/TXT/?qid=1411474765736&uri=CELEX:52012PC0011.

[8] Article 29 Data Protection Working Party. *Opinion 1/2010 on the concepts of "controller" and "processor"*. 2010. URL: http://ec.europa.eu/justice/policies/privacy/docs/wpdocs/2010/wp169_en.pdf.

[9] European Union. *DIRECTIVE 95/46/EC OF THE EUROPEAN PARLIAMENT AND OF THE COUNCIL of 24 October 1995 on the protection of individuals with regard to the processing of personal data and on the free movement of such data*. 1995. URL: http://eur-lex.europa.eu/legal-content/DA/TXT/PDF/?uri=CELEX:31995L0046&from=en.

[10] European Commision. *Factsheet on the "Right to be Forgotten" ruling (C-131/12)*. 2014. URL: http://ec.europa.eu/justice/data-protection/files/factsheets/factsheet_data_protection_en.pdf.

[11] Court of Justice of the European Union. *PRESS RELEASE No 70/14*. 2014. URL: http://curia.europa.eu/jcms/upload/docs/application/pdf/2014-05/cp140070en.pdf.

[12] Joris van Hoboken. *Case Note, CJEU 13 May 2014, C-131/12 (Google Spain)*. 2014. URL: http://ssrn.com/abstract=2495580.

[13] Article 29 Data Protection Working Party. *Guidelines on the implementation of the Court of Justice of the European Union. Judgment on "GOOGLE SPAIN AND INC V. AGENCIA ESPAÑOLA DE PROTECCIÓN DE DATOS (AEPD) AND MARIO COSTEJA GONZÁLEZ" C-131/12*. 2014. URL: http://ec.europa.eu/justice/data-protection/article-29/documentation/opinion-recommendation/files/2014/wp225_en.pdf.

[14] Peter Druschel, Michael Backes, and Rodica Tirtea. *The right to be forgotten - between expectations and practice*. 2012. URL: https://www.enisa.europa.eu/activities/identity-and-trust/library/deliverables/the-right-to-be-forgotten.

[15] Dan Boneh and Richard J. Lipton. *A revocable backup system*. In Proceedings 6th USENIX Security Conference, pp. 91–96. 2012. URL: http://static.usenix.org/publications/library/proceedings/sec96/boneh.html.

[16] R. Perlman. "File system design with assured delete". In: *Security in Storage Workshop, 2005. SISW '05. Third IEEE International*. Dec. 2005, 6 pp.–88. DOI: 10.1109/SISW.2005.5.

[17] SrijithK. Nair et al. "A Hybrid PKI-IBC Based Ephemerizer System". English. In: *New Approaches for Security, Privacy and Trust in Complex Environments*. Ed. by Hein Venter et al. Vol. 232. IFIP International Federation for Information Processing. Springer US, 2007, pp. 241–252. ISBN: 978-0-387-72366-2. DOI: 10.1007/978-0-387-72367-9_21.

[18] Nikolai Joukov, Harry Papaxenopoulos, and Erez Zadok. "Secure Deletion Myths, Issues, and Solutions". In: *Proceedings of the Second ACM Workshop on Storage Security and Survivability*. StorageSS '06. New York, NY, USA: ACM, 2006, pp. 61–66. ISBN: 1-59593-552-5. DOI: 10.1145/1179559.1179571.

[19] Roxana Geambasu et al. "Vanish: Increasing Data Privacy with Self-Destructing Data." In: *USENIX Security Symposium*. 2009, pp. 299–316. URL: https://www.usenix.org/legacy/events/sec09/tech/full_papers/sec09_crypto.pdf.

[20] Roxana Geambasu et al. "Comet: An active distributed key-value store." In: *OSDI*. 2010, pp. 323–336. URL: https://www.usenix.org/legacy/event/osdi10/tech/full_papers/Geambasu.pdf.

[21] C. Castelluccia et al. "EphPub: Toward robust Ephemeral Publishing". In: *Network Protocols (ICNP), 2011 19th IEEE International Conference on*. 2011, pp. 165–175. DOI: 10.1109/ICNP.2011.6089048.

[22] Scott Wolchok et al. "Defeating Vanish with Low-Cost Sybil Attacks Against Large DHTs." In: *NDSS*. 2010. URL: https://jhalderm.com/pub/papers/unvanish-ndss10-web.pdf.

[23] I. Clarke et al. "Protecting free expression online with Freenet". In: *Internet Computing, IEEE* 6.1 (2002), pp. 40–49. ISSN: 1089-7801. DOI: 10.1109/4236.978368.

[24] Ian Clarke et al. "A distributed decentralised information storage and retrieval system". PhD thesis. Master's thesis, University of Edinburgh, 1999.

[25] Ian Clarke et al. "Private communication through a network of trusted connections: The dark freenet". In: *Network* (2010).

[26] Ian Clarke et al. "Freenet: A Distributed Anonymous Information Storage and Retrieval System". English. In: *Designing Privacy Enhancing Technologies*. Ed. by Hannes Federrath. Vol. 2009. Lecture Notes in Computer Science. Springer Berlin Heidelberg, 2001, pp. 46–66. ISBN: 978-3-540-41724-8. DOI: 10.1007/3-540-44702-4_4.

[27] Stefanie Roos et al. "Measuring Freenet in the Wild: Censorship-Resilience under Observation". English. In: *Privacy Enhancing Technologies*. Ed. by Emiliano De Cristofaro and StevenJ. Murdoch. Vol. 8555. Lecture Notes

in Computer Science. Springer International Publishing, 2014, pp. 263–282. ISBN: 978-3-319-08505-0. DOI: 10.1007/978-3-319-08506-7_14.

[28] Wojciech Purczynski and Michal Zalewski. *Juggling with packets: floating data storage*. 2003. URL: http://lcamtuf.coredump.cx/juggling_with_packets.txt.

[29] K Rosenfeld, H.T. Sencar, and N. Memon. "Volleystore: A Parasitic Storage Framework". In: *Information Assurance and Security Workshop, 2007. IAW '07. IEEE SMC*. June 2007, pp. 67–75. DOI: 10.1109/IAW.2007.381916.

A Appendix A: Article 17 of the General Data Protection Regulation

This Appendix shows Article 17, Right to erasure, of the General Data Protection Regulation (GDPR). It concerns the text adopted by European Parliament for the 1st reading[45].

Article 17, Right to erasure:

1. The data subject shall have the right to obtain from the controller the erasure of personal data relating to them and the abstention from further dissemination of such data, and to obtain from third parties the erasure of any links to, or copy or replication of, that data where one of the following grounds applies:

> (a) the data are no longer necessary in relation to the purposes for which they were collected or otherwise processed;
>
> (b) the data subject withdraws consent on which the processing is based according to point (a) of Article 6(1), or when the storage period consented to has expired, and where there is no other legal ground for the processing of the data;
>
> (c) the data subject objects to the processing of personal data pursuant to Article 19[46];
>
> (ca) a court or regulatory authority based in the Union has ruled as final and absolute that the data concerned must be erased;
>
> (d)the data has been unlawfully processed.

1a. The application of paragraph 1 shall be dependent upon the ability of the controller to verify that the person requesting the erasure is the data subject.

2. Where the controller referred to in paragraph 1 has made the personal data public without a justification based on Article 6(1), it shall take all reasonable steps to have the data erased, including by third parties, without prejudice to Article 77[47]. The controller shall inform the data subject, where possible, of the action taken by the relevant third parties.

3. The controller and, where applicable, the third party shall carry out the erasure without delay, except to the extent that the retention of the personal data is necessary:

> (a) for exercising the right of freedom of expression in accordance with Article 80[48];
>
> (b) for reasons of public interest in the area of public health in accordance with Article 81[49];

[45] http://www.europarl.europa.eu/sides/getDoc.do?type=TA&language=EN&reference=P7-TA-2014-0212
[46] Right to object
[47] Right to compensation and liability
[48] Processing of personal data and freedom of expression
[49] Processing of personal data concerning health

(c) for historical, statistical and scientific research purposes in accordance with Article 83[50];

(d) for compliance with a legal obligation to retain the personal data by Union or Member State law to which the controller is subject; Member State laws shall meet an objective of public interest, respect the right to the protection of personal data and be proportionate to the legitimate aim pursued;

(e) in the cases referred to in paragraph 4.

4. Instead of erasure, the controller shall restrict processing of personal data in such a way that it is not subject to the normal data access and processing operations and can not be changed anymore, where:

(a) their accuracy is contested by the data subject, for a period enabling the controller to verify the accuracy of the data;

(b) the controller no longer needs the personal data for the accomplishment of its task but they have to be maintained for purposes of proof;

(c) the processing is unlawful and the data subject opposes their erasure and requests the restriction of their use instead;

(ca) a court or regulatory authority based in the Union has ruled as final and absolute that the data concerned must be restricted;

(d) the data subject requests to transmit the personal data into another automated processing system in accordance with paragraphs 2a of Article 15[51];

(da) the particular type of storage technology does not allow for erasure and has been installed before the entry into force of this Regulation.

5. Personal data referred to in paragraph 4 may, with the exception of storage, only be processed for purposes of proof, or with the data subject's consent, or for the protection of the rights of another natural or legal person or for an objective of public interest.

6. Where processing of personal data is restricted pursuant to paragraph 4, the controller shall inform the data subject before lifting the restriction on processing.

7. *Article 7 moved to Article 8a.*

8. Where the erasure is carried out, the controller shall not otherwise process such personal data.

8a. The controller shall implement mechanisms to ensure that the time limits established for the erasure of personal data and/or for a periodic review of the need for the storage of the data are observed.

9. The Commission shall be empowered to adopt, after requesting an opinion of the European Data Protection Board, delegated acts in accordance with Article 86 for the purpose of further specifying:

[50] Processing for historical, statistical and scientific research purposes
[51] Right to access and to obtain data for the data subject

(a) the criteria and requirements for the application of paragraph 1 for specific sectors and in specific data processing situations;

(b) the conditions for deleting links, copies or replications of personal data from publicly available communication services as referred to in paragraph 2;

(c) the criteria and conditions for restricting the processing of personal data referred to in paragraph 4.