



Maastricht University

KNOWLEDGE IN ACTION

## Faculteit Wetenschappen School voor Informatietechnologie

master in de informatica

### **Masterthesis**

***Priveasy - improving privacy infrastructure of existing web applications***

**Martijn Luyckx**

Scriptie ingediend tot het behalen van de graad van master in de informatica

### **PROMOTOR :**

Prof. dr. Kris LUYTEN

De transnationale Universiteit Limburg is een uniek samenwerkingsverband van twee universiteiten in twee landen: de Universiteit Hasselt en Maastricht University.



**UHASSELT**

KNOWLEDGE IN ACTION

[www.uhasselt.be](http://www.uhasselt.be)

Universiteit Hasselt  
Campus Hasselt:  
Martelarenlaan 42 | 3500 Hasselt  
Campus Diepenbeek:  
Agoralaan Gebouw D | 3590 Diepenbeek

**2020**  
**2021**



**Maastricht University**

# **Faculteit Wetenschappen**

## ***School voor Informatietechnologie***

master in de informatica

### ***Masterthesis***

***Priveasy - improving privacy infrastructure of existing web applications***

**Martijn Luyckx**

Scriptie ingediend tot het behalen van de graad van master in de informatica

### **PROMOTOR :**

Prof. dr. Kris LUYTEN



HASSELT UNIVERSITY

MASTER'S THESIS

---

**PRIVEASY — Improving privacy  
infrastructure of existing web applications**

---

*Author:*  
Martijn LUYCKX

*Advisor:*  
Prof. dr. Kris LUYTEN  
*Mentor:*  
Mariano DI MARTINO

*A thesis submitted in fulfillment of the requirements  
for the degree of Master of Computer Science*

*in the*

Expertise centre for Digital Media (EDM)  
Hasselt University

2020-2021





# *Abstract*

Expertise centre for Digital Media (EDM)  
Hasselt University

Master of Computer Science

**PRIVEASY — Improving privacy infrastructure of existing web applications**

by Martijn LUYCKX

When the world gave birth to the World Wide Web, privacy was not a concern. Through decades, the main principles have remained largely the same, leaving many applications on the web today with privacy and security as an afterthought. But can they be blamed? Regulations regarding privacy of personal information have just recently started to catch up with the mass amounts of personally identifiable information floating around on various places on the web. This thesis proposes two ways to improve the privacy infrastructure of those existing web applications.

First of all, a prototype was built that connects to an app's database and enables easy handling of GDPR rights requests. Specifically, right of access and right of erasure (also known as right to be forgotten). With that, the feasibility of building such a standalone tool that does not require any alterations to the application's source code is proven. The tool was validated by exploring an extensive example.

Secondly, when a privacy-related incident does happen (which is not a rare occurrence), this work tested the need for personalised information with the public. To do so, such a tool was built and tested for the Facebook phone number leak of April 2021, called *Ben Ik Erbij?*. The need for this personalised information was found to be high. The tool was reported on by national news outlets, it established a successful partnership with the Belgian Data Protection Authority on their initiative, and it performed an estimated six million look-ups in Belgium (population of 11.5 million).



## *Acknowledgements*

This thesis would not have been complete if I had to write this during isolation. Luckily, social distancing did not prevent me from consulting several people either in person or digitally.

First of all, I would like to thank my advisor Prof. dr. Kris LUYTEN for giving me the opportunity to work on a topic I am very interested in. In addition that, we have had many helpful brainstorming sessions and I have gotten very useful feedback from my advisor.

Secondly, I truly appreciate the help I got from my mentor Mariano DI MARTINO. Even though my thesis is outside of his research group, his knowledge about privacy, security and just general research were tremendously helpful. This thesis would not be complete if I didn't discuss both content and writing style of this work with him.

Thirdly, the mental support from both my peers and my family were essential to keep pushing through and finish this thesis in a qualitative way.

Lastly, my girlfriend deserves some special attention. She had to cope with my mental low points, she had to deal with my struggle finishing this work. Despite that, she kept supporting me immensely and helping me where she could.

Everyone I mentioned here can count on the fact that I genuinely and deeply appreciate their help or support in any way. Thanks everyone.





# Contents

<b>Abstract</b>	<b>iii</b>
<b>Acknowledgements</b>	<b>v</b>
<b>1 Introduction</b>	<b>1</b>
<b>2 Background</b>	<b>3</b>
2.1 Privacy	3
2.1.1 Importance	3
2.1.2 Privacy By Design	3
Principles	3
Related work: ISO	4
2.1.3 GDPR	4
General Regulations	5
Processing data	5
Fines	6
2.2 Data	6
2.2.1 Definition	6
2.2.2 Databases	7
Technologies	7
2.3 Content management systems	8
<b>3 GDPR Rights Toolkit</b>	<b>11</b>
3.1 Use-case	12
3.2 Requirements	12
3.3 Implementation	13
3.3.1 Constraints	13
3.3.2 Technical Properties	15
3.4 Validation	22
3.4.1 Sample employees database	22
3.4.2 Mock health sensor database	26
<b>4 Informing the public after an incident</b>	<b>29</b>
4.1 Types of privacy-related incidents	29
4.1.1 Hacking	30
4.1.2 Unauthorised disclosure	31
4.1.3 Theft	31
4.2 Case study: Facebook's global data leak in 2021	32
4.2.1 Timeline and communication	32
4.2.2 Compromised credential checking	34
Reasoning	34
Method of searching	34
Technical implementation	35

Results . . . . .	39
Belgian Data Protection Authority . . . . .	41
<b>5 Conclusion</b>	<b>43</b>
<b>Bibliography</b>	<b>45</b>

# List of Figures

2.1	A many to one relationship in a relational database. The <i>Customer ID</i> column in the <i>Orders</i> table is a foreign key to the <i>Customer</i> table. Image from FileMaker.com [3]. . . . .	8
2.2	Distribution of CMS market share. Data from Built With [70], formatted by Artem Minaev from First Site Guide [24]. . . . .	9
3.1	Database trends about (No)SQL usage [61] . . . . .	13
3.2	Database trends about different SQL technologies [61] . . . . .	14
3.3	Database trends about multi database usage [61] . . . . .	14
3.4	First step of the tool's setup: entering the database credentials . . . . .	16
3.5	Database schema of the employees sample database by MySQL [14] . . . . .	17
3.6	Second step of the tool's setup: selecting the main table for users . . . . .	18
3.7	Third (and last) step of the tool's setup: confirming that the found relations are correct . . . . .	19
3.8	Looking up users in the tool . . . . .	20
3.9	Taking action on a user's data in the tool . . . . .	21
3.10	The tool shows most tables exactly how you would expect . . . . .	25
3.11	The tool separates some data into different and/or multiple views as a result of the prototyping method used . . . . .	25
3.12	A (redacted) snippet of the information presented by the tool for Flow Connect . . . . .	27
4.1	Graph depicting yearly occurrences of the kinds of data breaches by Enisa [5] . . . . .	30
4.2	Facebook's response to the report by Inti De Ceukelaire [8] . . . . .	33
4.3	Results of the benchmark on different technology stacks to perform phone number lookups in a database . . . . .	37
4.4	Left: unaltered network communication of Ben Ik Erbij? - Right: Schematic depiction of a data anonymisation method . . . . .	38
4.5	A screenshot of the tool ' <i>Ben Ik Erbij?</i> ' . . . . .	39
4.6	Total requests to <i>Ben Ik Erbij?</i> per country based on domain TLDs as counted by Cloudflare . . . . .	40
4.7	Unique visitors to <i>Ben Ik Erbij?</i> per country based on domain TLDs as counted by Cloudflare . . . . .	40
4.8	CPU usage peaks of <i>Ben Ik Erbij?</i> 's server corresponding to live news reports based on Google Cloud's analytics page . . . . .	41



## Chapter 1

# Introduction

When the web was built more than three decades ago, privacy was not a concern. The initial purpose of the world wide web (WWW) was to share documents between research institutions freely. Those parties trusted each other, so privacy and security were not the default. As the web has been used increasingly more over the years for more varying applications, the amount of (personal) data accessible for everyone has increased as well. This brought a need for secure applications with privacy in mind.

One of the reasons that privacy is so important on the web nowadays is malicious hackers, individuals or groups of actors whose intention it is to earn money by conducting illegal activities on the web. A common way for them to extract money from victims is phishing (or other types of social engineering), which are most effective when attackers can work with as much personal info of the victim as possible to gain trust.

Regulators have been catching up in the past years. The European General Data Protection Regulation (GDPR) and the Californian Consumer Privacy Act (CCPA) are two examples of government instances obligating businesses to be more mindful of personal data.

Unfortunately, it is virtually impossible to provide a one-size-fits-all solution to fully comply with all regulations, as every application is different. Additionally, a large portion of the existing web applications will not have the means (budget, human resources) to build a custom solution for their legacy application.

Since this work aims to improve that situation, it will attempt to answer two research questions:

- Is it possible to create a solution to (partly) standardise compliance with (a part of) privacy regulations?
- Is it effective to inform the public in a personalised way when a data leak has occurred?

The first research question will be answered as a feasibility study. A prototype will be attempted to be built to cover the largest scope possible with the least amount of friction to actually implement it in an existing web application. The second research question is handled in the form of a case study. A tool will be built to inform the public about a specific data leak, and the results will be analysed.

In the next chapter, the concepts around privacy, data, and their importance will be detailed, as well as related work in the space. Chapter 3 will research the feasibility of building a tool that can help existing web applications to comply with a part of the GDPR regulation as frictionless as possible. After that, Chapter 4 analyses the tool's results that can help existing web applications inform the public in case of a data leak.

## Chapter 2

# Background

### 2.1 Privacy

The Oxford English Dictionary defines privacy as *"a state in which one is not observed or disturbed by other people"* and *"the state of being free from public attention"*. Notice how that differs from secrecy. Wanting privacy does not mean one has something to hide. Instead, it promotes personal control of what and when information is revealed to someone or the public. This control is essential to freedom and human rights in general, so much so that Article 8 of the Human Rights Acts states that "everyone has the right to respect for his private and family life, his home and his correspondence" [21].

#### 2.1.1 Importance

It may seem intuitive to the reader that privacy is important and becoming even more important than it ever was in this digital era. Consumers International has quantified those intuitions. They conducted a survey asking participants which area they felt had the greatest need for a new standard. The results spoke for themselves: *"Commercial collection, storage and use of personal data"* was voted in the personal top three of 53% of participants. It was the first choice of 26% of all voters. Closely related, *"Security of connected products (Internet of Things)"* was voted the second most important area [48].

#### 2.1.2 Privacy By Design

Privacy by design is an approach to data regulation and privacy that advocates making privacy the default. It was introduced by Ann Cavoukian in 2010 and later formalised by renowned institutions [7, 35]. There have been major efforts regarding Privacy by Design already. One initiative is the European Union's regulation: the General Data Protection Regulation, or GDPR for short [27]. The International Organization for Standardization (will be referred to as ISO from now on) is also working on an official standard for Privacy by Design [38]. Their new committee, ISO/PC 3171, goes about consumer protection and aims to bring the certificate "privacy by design for consumer goods and services" to the public [39]. The following sections will take a look at some of these efforts regarding this topic.

#### Principles

For privacy, personal control and freedom of choice are key. The principles of Privacy by Design reflect that [12], stating the following:

- Proactive, not reactive



- Privacy as the default setting
- Privacy embedded in the design
- Full functionality, positive-sum
- End-to-end security, full life cycle protection
- Visibility and transparency (keep it open)
- Respect for user privacy (keep it user-centric)

The item needing most explaining is "full functionality, positive-sum. Ann Cavoukian contrasts positive-sum and zero-sum as "replace 'vs' with 'and'" [7]. Privacy by Design aims not to have to choose between values, but combine them all. For example, Ann believes that improving security does not imply sacrificing privacy. On the contrary, they can and should go hand in hand and be developed with both interests simultaneously.

### Related work: ISO

The International Organization for Standardization, ISO for short, is working on an internationally recognised standard for Data Privacy By Design [38]. At the time of writing, it is being drafted ISO/PC 317 "Consumer protection: privacy by design for consumer goods and services". In their news article about the effort for a new standard, they explain that "the new ISO project committee, ISO/PC 3171), Consumer protection: privacy by design for consumer goods and services, was developed by ISO/COPOLCO, the ISO committee that deals with consumer issues in standardization" [38, 39].

ISO explains the need for a new standard with a fitting analogy [47]. You get on an aeroplane, they state, because you trust it. You did not inspect the engine, and you did not get out your screwdriver and tighten the seems. You trust it because there are standards. An aeroplane is not allowed to fly without going through an inspection and complying with an internationally recognised standard.

Right now, you can use platforms, and you can enjoy platforms. You can gain benefit from platforms. But right now, most do not trust them with their data [47]. ISO wants the same level of trust for data as people have for aeroplanes.

The way to get there is to be safe by default. Simplifying terms and conditions, or even shortening them, will not help, ISO states. People want technology to move so quickly. They do not spend the time to read any terms and conditions. This should not have less privacy as a consequence, they argue. Privacy should be the default option, with positive consent required to reveal or share more information.

### 2.1.3 GDPR

Since its introduction on May 25, 2018, GDPR has greatly affected many businesses that operate with customer data. Most notably: every company that operates with data from EU citizens must now comply with many subtle and less subtle regulations that may change the interaction with the user and the usage and storage of personal data [27, 59]. The GDPR calls itself "the toughest privacy and security law in the world" [27].

## General Regulations

The GDPR tightens the previously existing Data Protection laws and regulations. Most notably, it imposes more obligations and accountability for all parties involved in processing any personal information. The regulation attacks the problem of privacy as a default with the requirement of positive consent. The consumer also benefits from more rights, like the right to data access and the right to be forgotten [27, 59].

When a data breach happens, businesses that operate in regions covered by the GDPR are now obligated to report it. National authorities must know about every data breach that may affect customer privacy within 72 hours [69].

## Processing data

When processing data, you must do so with respect to the seven data processing principles of the GDPR [27, 37].

1. Lawfulness, fairness and transparency
2. Purpose limitation
3. Data minimization
4. Accuracy
5. Storage limitation
6. Integrity and confidentiality
7. Accountability

The following paragraphs will explain these seven principles in more detail, in chronological order.

Treating personal data with fairness and transparency ensures that there are no unexpectancies when it comes to people's data. Evidently, making sure that no data is treated unlawfully aids that cause. Users should feel like they know why they are giving data to a third party and should not be surprised by usage outside of those expectations.

The party collecting the data should explain every purpose of the data *before* said data is collected from a person. Those purposes should also be limited to explicit ones, in contrast to collecting data for generic analysis or archiving.

If the limited useful purposes for data collection have been identified, it is key to also limit the collected data to only those data points which are relevant to the purpose. Additionally, the GDPR states that it is important that those data points are accurate and are kept accurate by updating them (to the point of usefulness for the purpose).

More and more important decisions are made based on personal data stored in digital systems. That is why the GDPR emphasises the importance of keeping that data accurate, complete and up-to-date. That way, those systems can make decisions based on accurate and complete information.

When a purpose is no longer relevant, the data which was collected for that purpose should be destroyed. Since the purpose of collected data needs to be stated before the actual collection (as mentioned previously), merely finding another use for data does not warrant extended storage. The reason for that being the longer data is stored on a system, the greater the risk of that data getting into the wrong hands or for it being used unlawfully.

As data is collected for a specific purpose, parties must make reasonable efforts to make sure that no one can access, delete or modify the data unrelated to that purpose. Expected measures range from physical (if applicable) to organisational and technical. Those measures help to ensure the integrity and confidentiality of the collected personal data.

Lastly, GDPR states that every party (controllers and processors) have to work towards compliance with these principles and can be held accountable if they do not succeed. The parties must also be able to demonstrate which measures are taken to reach that compliance.

### Fines

To very clearly emphasise the importance of privacy, violators of the GDPR may be fined up to €20 million or up to 4% of the annual worldwide turnover of the preceding financial year in case of an enterprise, whichever is greater. The EU states that *"the GDPR's stiff fines are aimed at ensuring best practices for data security are too costly not to adopt"* [26].

## 2.2 Data

This work talks a lot about data, customer data, sensitive data and so forth. It is important to clearly define what is meant by those terms to eliminate ambiguity in those statements.

Additionally, the upcoming sections will discuss how modern digital systems store said data and touch upon different methodologies. As you may know, storing data in different ways can have a big impact on performance when working with large amounts of data. It is, however, important not overlook to semantic implications when storing data a certain way.

### 2.2.1 Definition

According to the OECD Glossary of Statistical Terms, *"data are characteristics or information, usually numerical, that are collected through observation"* [57]. More specifically, the book states that personal data (in literature sometimes called 'personally identifiable information' [62]) is *any information relating to an identified or identifiable natural person ('data subject')*. It clarifies that an identifiable person is *one who can be identified, directly or indirectly*. That definition is later contrasted with anonymous data, where the individual is not identifiable based on the data [57].

### 2.2.2 Databases

Digital applications often need to store and/or access (personal) data. A system that allows one to do that is called a database. Often, the term database refers to a collection of data managed by a *database management system* (or DBMS) [16].

According to Database Systems: The Complete Book, a DBMS is expected to [16]:

- Allow users to create new databases
- Give users the ability to query the data
- Support the storage of very large amounts of data
- Have the ability to recover data the face of failures
- Control access to data from many users at once (without allowing unexpected or partial interactions)

### Technologies

Modern databases can be organised into two major categories:

- (1) Relational databases
- (2) NoSQL databases

Although it is fair to mention that older types of databases paved the way for these modern standards. In the past, plain files were used as a storage medium [28]. Many important files on Linux (or other Unix-like systems) still utilise this (for example, the `/etc/passwd` file). Also, CSV (comma-separated values) files are still a popular export format for many systems including ever-popular spreadsheet applications.

NoSQL databases can be further split up into three categories:

- (2a) Key-value stores
- (2b) Document stores
- (2c) Graph databases

As the name implies, relational databases are a type of database which is based on a relational model of data. Specifically on the relational model of data as proposed by E. F. Codd [11]. For the purposes of this work, it is especially relevant that this type of database stores keys and foreign keys to establish relations. A many-to-one relation is depicted in Figure 2.1, where the *Orders* table has a column *Customer ID*, which is a foreign key. That is, the column references the unique identifier of another column to establish the relation.

Thanks to the fact that modern database systems can enforce this foreign key constraint, it is also possible to query the relationships themselves, which will prove useful in Chapter 3.

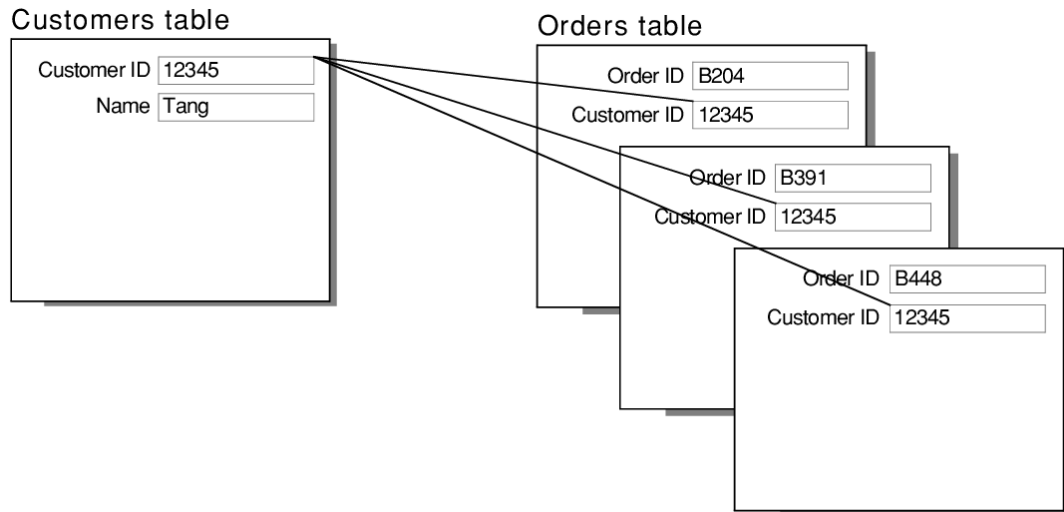


FIGURE 2.1: A many to one relationship in a relational database. The *Customer ID* column in the *Orders* table is a foreign key to the *Customer* table. Image from FileMaker.com [3].

## 2.3 Content management systems

Many of the world's websites run on some kind of content management system (CMS). According to *Built With*, that exact number is 65,474,797 websites that are built with a CMS as of August 4, 2021 [70]. Of those millions of websites, 42% use WordPress. The second-largest share is for Wix, with a 7% market share. More of the market share distribution can be seen in Figure 2.2.

Unfortunately, there is a big difference between these two most popular CMS systems: the smaller one (Wix) has a built-in system for complying with GDPR rights [72], but the larger one (WordPress) does not have such a system [73].

Wix offers a support article to its users, titled "*Preparing your Wix site for GDPR*" [71]. At the very bottom of the page, it mentions, "*Use Wix Tools to Access and Delete Your Site Visitors' Data*". The entire section is short enough to quote here:

*"In accordance with GDPR, site-visitors have the right to access their data or 'be forgotten' (be permanently deleted from your databases). Wix has developed tools to assist you in becoming GDPR compliant:*

- *Right to access. Learn more.*
- *Right to be forgotten. Learn more."*

The fact that this section is so brief is not a bad thing. In fact, the *learn more* hyperlinks make the user navigate to a different page all about *customer's data files* [72]. Wix has built a system where a website administrator can get a copy of their customer's data file or permanently delete it.

In contrast, for the numerous websites which are built on top of WordPress, they have to rely on third-party plugins [36]. Taking into account that a large portion of the web does not make use of a CMS, it is easy to imagine that many websites do not have a streamlined way (if any at all) to handle GDPR rights requests.











<b>CMS Technology</b>	<b>Live Websites</b>	<b>Market share</b>
 WordPress	28,183,568	43.41%
 Wix	4,565,423	7.03%
 Squarespace	2,750,270	4.24%
 Joomla!	1,580,832	2.44%
 GoDaddy Website Builder	1,624,154	2.50%
 Weebly	1,019,509	1.57%
 Duda	705,120	1.10%
 Blogger	672,560	1.05%
 Drupal	562,655	1.03%
 Jimdo	486,843	0.76%

FIGURE 2.2: Distribution of CMS market share. Data from Built With [70], formatted by Artem Minaev from First Site Guide [24].



## Chapter 3

# GDPR Rights Toolkit

This chapter will explore technologies to develop a tool that provides a way to handle requests from users about their GDPR rights semi-automatically, arriving at a working prototype that proves the technical possibility of implementing this in the wild. After a one-time-only few-minute-long setup process, one can look up users and take actions on their data with the click of a button.

Current implementations are done a case-by-case basis companies that have a large budget and desire to do so. On some popular social media sites, you can even download all data related to you in an accessible format. An example of this is Facebook's "download a copy of your information" feature [23]. Evidently, this is built by Facebook themselves and is tailor-made to their platform and data structure. Smaller business do not have access to something like that.

A tool like this should help businesses comply with some of GDPR's seven data processing principles which were described in Section 2.1.3:

- **Lawfulness, fairness and transparency:** as businesses will be able to export all personal data related to a person very easily, people's GDPR requests to be informed, right of access and right to data portability can be taken care of at a moment's notice.
- **Accuracy:** since people can easily retrieve all data which is stored about them, they can also notify the relevant business about any inaccuracies in that data. At the same time, they can provide accurate, up-to-date information to replace it with.
- **Storage limitation:** When a person revokes the consent to process or store a specific piece of information, it can no longer be stored by the business. In the case of a person revoking all permission to store anything, this tool will help a business to identify and erase all relevant data for that person. In case of a partial deletion, both the business and person can use the retrieval functionality of the tool to verify that the information no longer shows up.
- **Accountability:** Since using this tool is a measure to help with compliance towards GDPR, it can be used by accountable parties to prove they have taken appropriate measures to implement these principles in practice.

The next sections will describe the process of engineering this toolkit. Also, they will tackle the technical challenges that arose.



### 3.1 Use-case

Many existing web applications, particularly small to medium-sized custom-built apps, were built (way) before strict privacy regulations like GDPR and are thus lacking in compliance. Think of a local restaurant with an online order system, that one home decoration store that had a webshop before it was cool, or the barbershop which has recently added the ability to book an appointment through their website.

Essentially, the websites which fall in the middle category (b) in this listing have the highest potential of having no system to comply with GDPR rights requests:

- (a) Websites made with no-code systems or website builders
- (b) Websites custom built by small, inexperienced teams (often chosen by smaller or local businesses for reduced cost and less overhead, but very custom functionality)
- (c) Websites constructed by large, experienced agencies

This chapter aims to explore the technical possibility of building a system that plugs right into those existing systems, to add a semi-automatic way of handling incoming GDPR rights requests (like the *right of access* and *right to be forgotten*).

### 3.2 Requirements

For a system as described in Section 3.1 to be of maximal utility, a few requirements need to be imposed.

- A. The system needs to be able to be plugged into existing applications without having to alter those applications.
- B. The setup needs to be able to be self-hosted to avoid having to send personally identifiable information to a third party.
- C. Once the setup has been completed, a user should be able to handle at least *the right of access* and *right to be forgotten* GDPR requests in an intuitive way.

First of all, it is important that this system will be standalone. If an application would need to take this toolkit into account from the get-go, not a single existing application will be supported. Since the goal is to help those existing applications, it is of crucial importance that this toolkit really will be built as an external add-on without any relation to the code base of the application.

Secondly, the finished toolkit (and any version before, for that matter) should be self-hosted. Self-hosting is necessary not to require trust from the user. Even for this prototype, testers should not be having to trust sending critically sensitive information to a third party. Evidently, the same goes for a published version. Having the toolkit on the server of the application avoids the issue of sending database credentials and PII to a third party (and avoid sending them over a network connection at all).

Thirdly, once the one-time setup has been completed, no technical knowledge should be required at all. This makes it possible that handling any GDPR rights request can in theory be possible by performing two steps:

1. Search for the user who made the request
2. Execute the appropriate action on that user

### 3.3 Implementation

Gauging hypothetical possibility of a toolkit such as described above can be done by building a prototype. By focusing on the technical difficulties, a minimal viable product (MVP) can be crafted without too much overhead.

#### 3.3.1 Constraints

A prototype, by its very nature, restricts the scope of a solution to some core aspects. In this case, some constraints were chosen on the environment in which this toolkit will operate. However, it is crucial that the restrictions do not limit the testing to a tiny or extremely niche scope. As in doing so, the prototype would not be an accurate representation of the technology to which it is helping to give birth. With that in mind, these are the constraints of this prototype.

- Restricted to MySQL
- Restricted to a single database
- One-time-setup is session-based
- Little concern for UI and UX

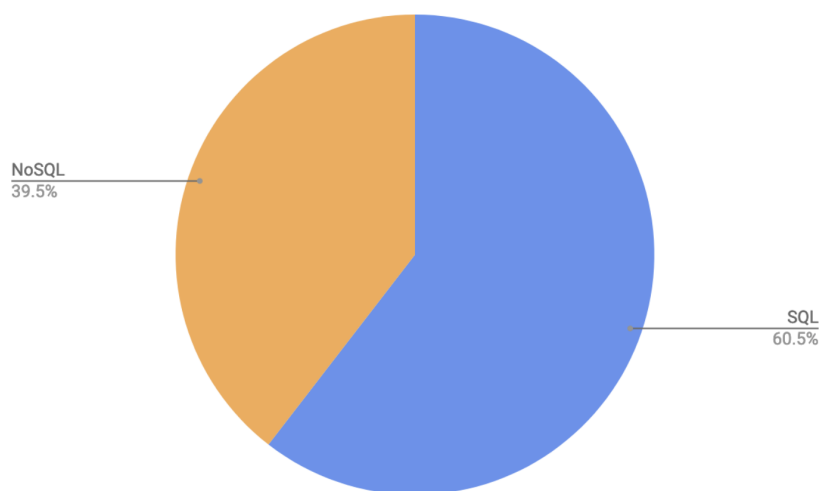


FIGURE 3.1: Database trends about (No)SQL usage [61]

The choice to restrict the prototype to a single database technology comes rather naturally. Supporting more than one implementation of the same database paradigm consumes lots of time and energy only to gain a larger testing area. In this case, MySQL was chosen. There are three reasons that support this.

- Relational databases are dominant [61] (Figure 3.1)
- MySQL is the most popular relational database technology [61, 19] (Figure 3.2)
- The author of this work is familiar with MySQL

These factors combine for an ideal scenario in prototype development. Firstly, since the author is familiar with the technology, a working MVP can be developed as quickly as possible. Secondly, since MySQL is the single most used database technology in the dominant relational database paradigm, the testing area is extensive.

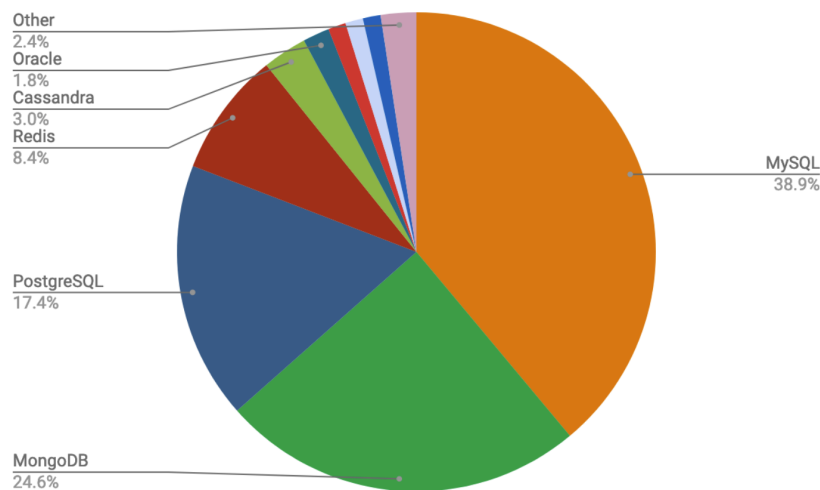


FIGURE 3.2: Database trends about different SQL technologies [61]

Restricting this prototype to a single database application also thins out the testing area in favour of faster development, as is the case with most (if not all) of these constraints. It is sensible, though, since the vast majority of users build their applications with a single database [61], as can be seen on Figure 3.3.

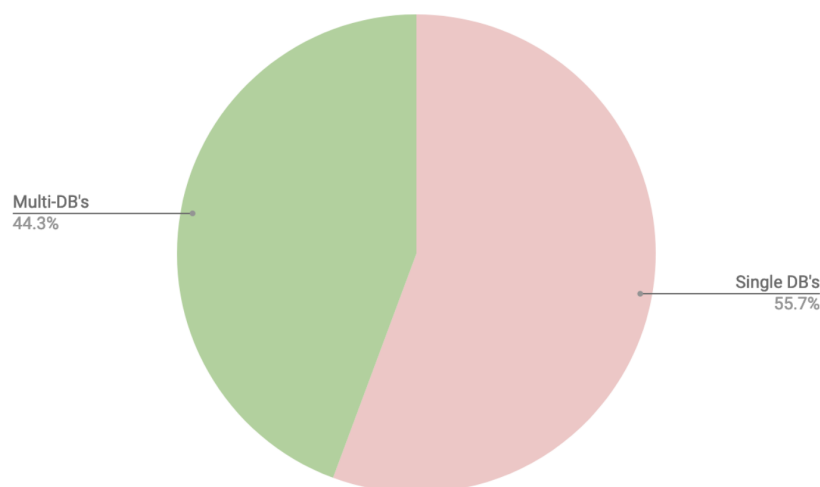


FIGURE 3.3: Database trends about multi database usage [61]

Building this prototype with only session state accomplishes two things. Firstly, it removes the need to use a long-term storage medium (like a database). This allows for a shorter development time. Secondly, it gives an incentive to craft the setup process in such a way that it is easy to go through. If the setup would be cumbersome, that would hinder the prototyping phase and lose a lot of time when testing.

Lastly, there is little concern for UI and UX. That does not mean no concern (especially for UX), but just enough to convey its functions clearly without spending too much time and effort on looks and experience of using the toolkit.

All of this accomplishes the same thing: get a working version of the toolkit ready as soon as possible, to gauge the theoretical possibility of building a fully featured in the future.

### 3.3.2 Technical Properties

The first step of the setup process is hosting the tool. Following the aforementioned requirement A, this has to be able to be done by the owner of the existing application, on a place they desire.

Since the tool has been written in PHP, the environment must satisfy these conditions:

- The server must be able to run PHP
- The PHP version must support MySQL
- The database must accept connections from the server

If all these requirements are met, the tool can be deployed. A handful of PHP files can be served without modification or need for configuration files.

When navigating to the tool, a user is met with the first setup: entering the database credentials. Keep in mind that this setup may be repeated in the future, as this prototype only stores the setup in a session as described in Section [3.3.1](#).

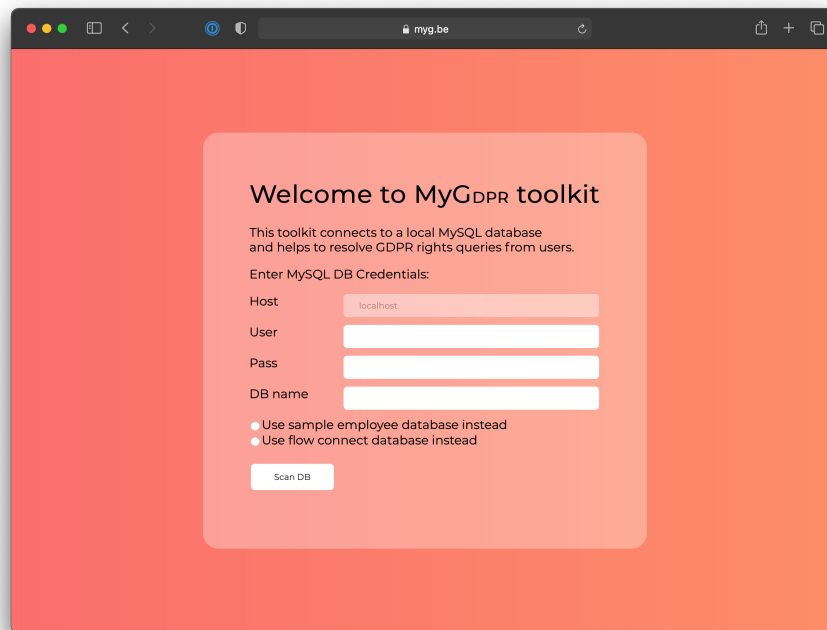


FIGURE 3.4: First step of the tool's setup: entering the database credentials

As can be seen in Figure 3.4, the interface is really straight forward. For testing purposes during this work, two sample databases with fake information have been added as an option. All screenshots from the tool in this work will utilise the sample employees database, provided by MySQL [14]. The schema of this sample database can be seen in Figure 3.5.

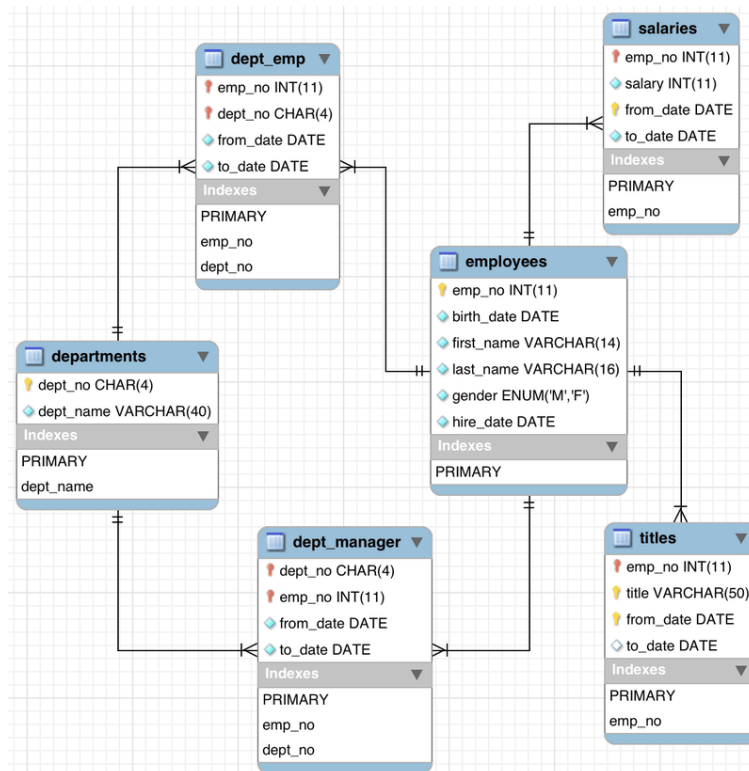


FIGURE 3.5: Database schema of the employees sample database by MySQL [14]

Naturally, this tool aims to work with personal data (PII). However, it is impossible for the software to infer semantic information reliably. Thus, it is unable to know which table contains the (unique identifier for) users. It follows then that the second step in the tool is to select which table contains that user data, as can be seen in Figure 3.6.

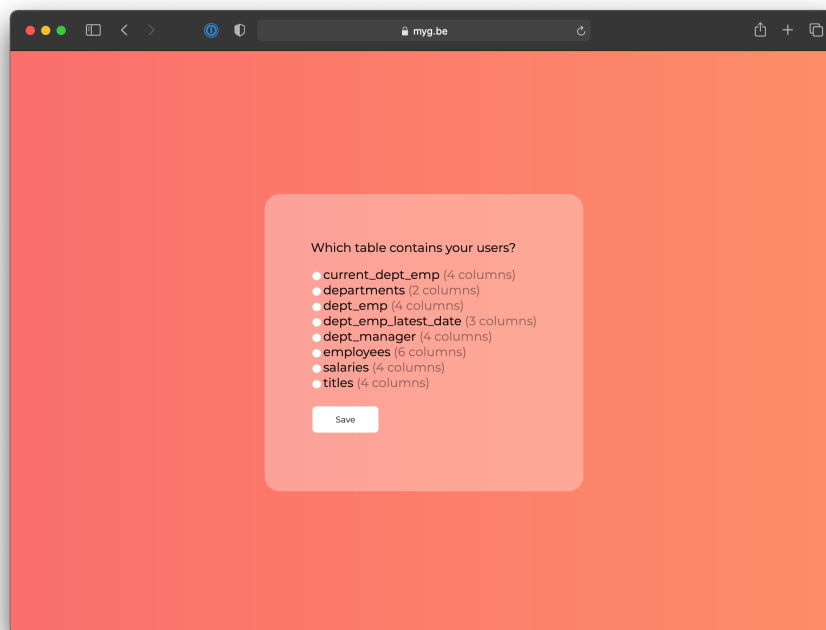


FIGURE 3.6: Second step of the tool's setup: selecting the main table for users

The tool queries the MySQL host to ask for the existing databases, which is simple:

```
SHOW DATABASES
```

From those results, it looks for the main database. Since this prototype restricted itself to single-database applications (see Section 3.3.1), it suffices to filter out the default tables that show up for MySQL systems. Those are the following [17]:

- `mysql`
- `information_schema`
- `performance_schema`
- `sys`

The tool then selects the database that remains. If there are multiple, it picks the first one from default ordering. This behaviour can be overridden by providing a database name in the first setup screen.

When it has identified the relevant database, it queries for the tables in there. That is where the default databases come in handy: that information is stored in the table `information_schema` by MySQL itself. The query looks as follows:

```
SELECT table_name, column_name, column_key
FROM information_schema.columns
WHERE table_schema = :dbname
ORDER BY table_name, column_key DESC, column_name;
```

The last part of the setup is confirming the database has its foreign key constraints correctly set up. To do so, the tool queries even more meta information from the `information_schema` table:

```
SELECT DISTINCT a.table_name           AS "FROM_TABLE",
               a.column_name          AS "FROM_COLUMN",
               a.referenced_table_name AS "TO_TABLE",
               a.referenced_column_name AS "TO_COLUMN",
               b.delete_rule
FROM   information_schema.key_column_usage a
      JOIN information_schema.referential_constraints b USING (constraint_name)
WHERE  a.referenced_table_schema = :dbname
      AND a.referenced_table_name = :tablename
      AND a.referenced_column_name = :primarykey
ORDER BY a.table_name asc;
```

As can be seen in the query, `information_schema` stores those foreign key constraints in `referential_constraints`. By joining that table with the information about all the columns and their keys (`key_column_usage`), it can be worked out which table has a foreign key relation with another. This has to be repeated in a slightly altered query for incoming and outgoing foreign key relations.

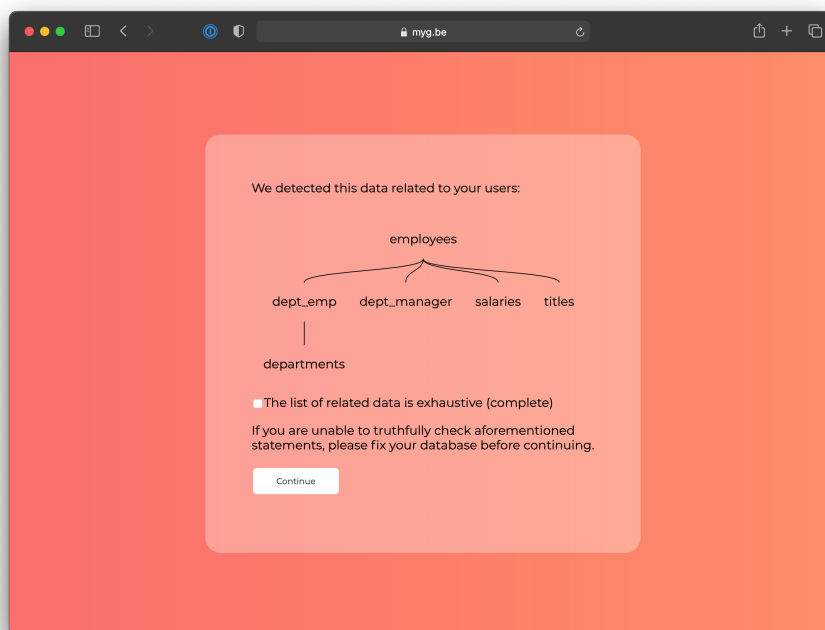


FIGURE 3.7: Third (and last) step of the tool's setup: confirming that the found relations are correct

To get all the information that is linked to a user, the tool starts searching from the only table it knows anything about semantically: the user table. Incoming and outgoing foreign key constraints are queried for that table. That will result in a list of new tables, one level separated from the user table.



Those new tables are then in turn queried for any foreign key relations both ways, giving next level of separation. This way, one-to-many and many-to-many relations have been accounted for and can be visualised in the graph-like way as shown in Figure 3.7.

The user which is responsible for the setup of the tool needs to confirm that these foreign key constraints in the database are present and correct. To be able to do so in an easy way, the tool shows the relations visually in a graph-style representation, as shown in Figure 3.7.

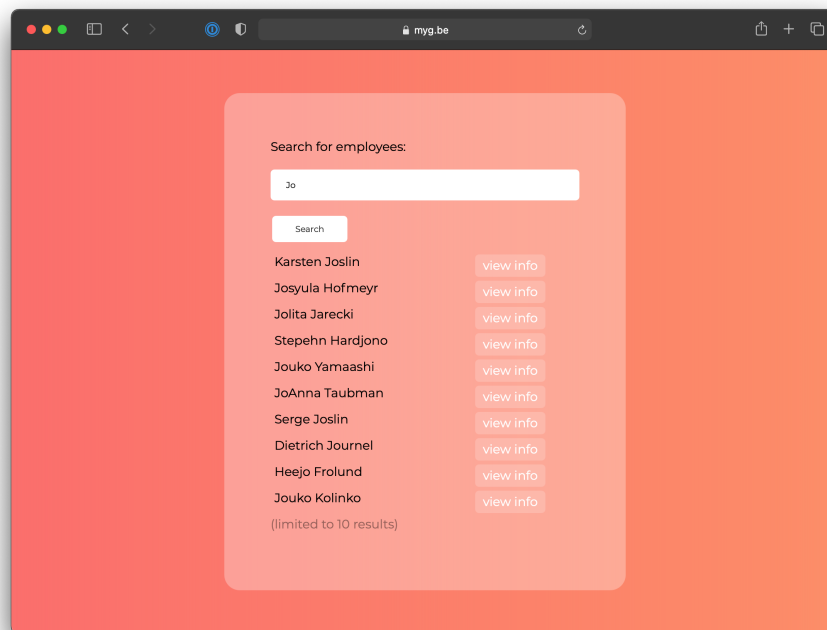


FIGURE 3.8: Looking up users in the tool

When the setup has been completed, it will be stored in a session to be re-used. The only thing that is left, is for the end user to search for the person they want to handle a GDPR rights request of, and to take the appropriate action.

The query to look up users searches on all columns, as the tool does not have any semantic information about the columns. To allow for user friendly searching, the search term is split up on white space (spaces) and then searched for in every column. A search term like *"John Doe"* would then work for a table which has first name and last name in separate columns, whereas no results would be returned if the full search term (*"John Doe"*) needed to match a single column.

For example, searching for `Daisy Critter` in the sample employee database (feel free to refresh your memory of the schema in Figure 3.5) results in the following query being executed:

```
SELECT *
FROM `employees`
WHERE `emp_no` LIKE :emp_no0
```

```

OR    `emp_no` LIKE :emp_no1

OR    `birth_date` LIKE :birth_date0
OR    `birth_date` LIKE :birth_date1

OR    `first_name` LIKE :first_name0
OR    `first_name` LIKE :first_name1

OR    `gender` LIKE :gender0
OR    `gender` LIKE :gender1

OR    `hire_date` LIKE :hire_date0
OR    `hire_date` LIKE :hire_date1

OR    `last_name` LIKE :last_name0
OR    `last_name` LIKE :last_name1
LIMIT 10;

```

In the query, all parameters are assigned the corresponding index of the search terms as their number after the column name. So `:column_name0` is a parameter for `Daisy` and `:column_name0` is a parameter for `Critter` in this example (where `column_name` is a placeholder for all the individual names of the columns).

This method was chosen instead of using a single parameter name for each search keyword because of a technical limitation of the technology stack used. It is not possible to use a single placeholder (e.g. `search_term0`) multiple times.

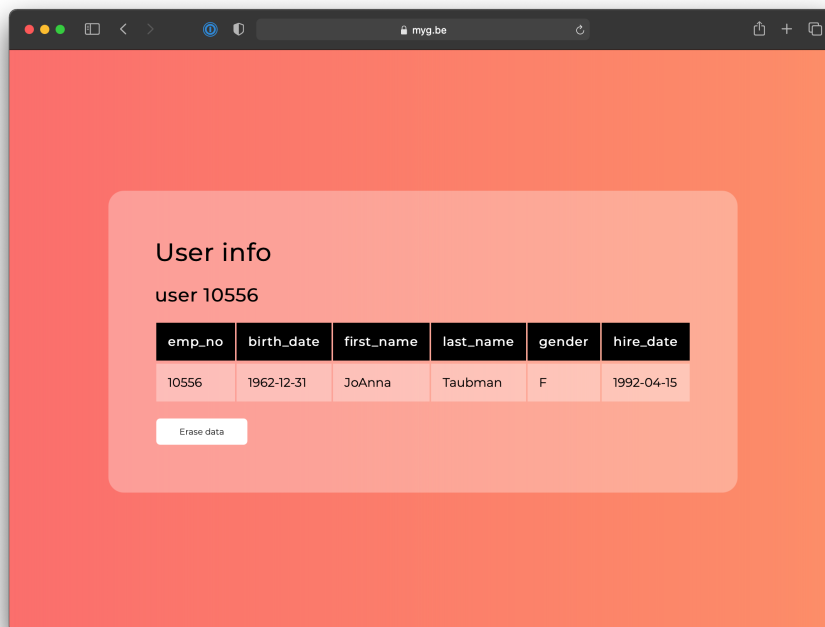


FIGURE 3.9: Taking action on a user's data in the tool

The last step is taking actions on the user's data, which is surprisingly straightforward after the setup process. All it takes to display the data is query all columns

from all tables related to the user, which were discovered during the phase of finding all tables with foreign key constraints (Figure 3.7).

Erasing data from a user is also easy: execute a `DELETE` statement on the user's ID, and cascade delete rules will take care of the rest (as described in the *Referential Actions* section of MySQL's docs on foreign key constraints [13]).

## 3.4 Validation

To validate that the tool works as expected, two examples from different databases will be run through the tool. To start, it will be described how one would expect the tool to work. Afterwards, it can be validated that the results match the expectations.

### 3.4.1 Sample employees database

The first example will be run on the sample employees database. The schema of the database can be seen in Figure 3.5. This database is built and provided by MySQL [14]. In the schema graph, you can see that the table which contains unique identifiers for users is `employees`. Notice that there are four tables connected to the `employees` table via relations (foreign key constraints):

- `dept_emp`
- `dept_manager`
- `titles`
- `salaries`

The only table not directly connected through relations to the `employees` is `departments`. This table is connected to both the `dept_emp` table and the `dept_manager` table.

As can be seen in Figure 3.7, the tool correctly understands this relationship structure through querying the foreign key constraints. A small side effect of the representation method is that it does not depict the fact that the `departments` table is also connected to `dept_manager` table.

Let's explore the actual semantics of these tables. The `employees` table has a few columns of obviously personally identifiable information and some information that only relates to the person (but isn't necessarily PII):

- `birth_date`
- `first_name`
- `last_name`
- `gender`
- `hire_date`

The first three (birth date and full name) are clearly PII. Gender and hire date are not considered PII, but are only applicable to that specific person and should therefore be removed when complying to a GDPR right to erasure request.

Data from the `titles` table is in a many-to-one relation with people in the `employees` table. Each title is for a specific employee during a specific period of time. While it could be argued that a specific sequence of title changes in a period of time could be used to identify a person, that is less of a concern in a large database that would store such information in the first place. It is not in the scope of this work to argue any way or the other. For this database, the deletion rule of the foreign key constraint will be followed. This opts for the deletion route (instead of the anonymisation route).

Similarly to the `titles` table, the `salaries` table has a salary column with additional columns to indicate a specific time frame (in the example data, that's usually a yearly salary). Also for this table, all rows related to an employee will be entirely deleted when requested.

The `dept_manager` table and the `dept_emp` tables are identical syntactically and very similar semantically, so they will be discussed as one. The difference between the tables can be inferred from their names: the former indicates that an employee was a manager for a certain department in a time frame, while the other indicates a person is an employee for a department. This information should be deleted on request as it is information directly and uniquely tied to a person.

Lastly, the `departments` table contains solely the name of each department. It is correctly detected as relating to an employee by the system, but should not be deleted when removing a user's information.

Summarising that semantic information, it is expected that when a user's information is deleted, the following is erased from the database:

- The row about the user in the `employees` table
- Every row from the `salaries` table that contains a foreign key to the employee's unique ID.
- Every row from the `titles` table that contains a foreign key to the employee's unique ID.
- Every row from `dept_emp` table that contains a foreign key to the employee's unique ID.
- Every row from `dept_manager` table that contains a foreign key to the employee's unique ID.

Additionally, it is expected that not a single row from the `departments` table is removed or altered.

When all information of a user is queried, it is expected that these same rows are shown instead. Also, the information about the department should be included as well. Even though it is not strictly information about the user, it is information that the company has about that person.

To now validate if the expected behaviour is performed by the tool, a specific example will be laid out in the next paragraphs.

The example will be about the fictive person *Marla Brendel*, a man born in June 1959 and hired in October 1985.

When Marla requests to access all the data his employer stores about him, he would expect to receive this data:

**Personal information:**

Birth date	First name	Last name	Gender	Hire date
1959-06-17	Marla	Brendel	M	1985-10-14

**Salary information:**

Salary	From date	To date
56720	1985-10-14	1986-10-14
60264	1986-10-14	1987-10-14
63968	1987-10-14	1988-10-13
66772	1988-10-13	1989-10-13
68160	1989-10-13	1990-10-13
71280	1990-10-13	1991-10-13
74279	1991-10-13	1992-10-12
74378	1992-10-12	1993-08-10

**Title information:**

Title	From date	To date
Senior Staff	1992-10-14	1993-08-10
Staff	1985-10-14	1992-10-14

**Employee at departments:**

Department	From date	To date
Finance	1985-10-14	1988-09-02
Human Resources	1988-09-02	1993-08-10

When Marla requests to be forgotten (right to erasure), this exact information should be removed.

Actually performing these actions through the tool yields positive results. Marla's data is displayed to the user nicely, and all the data one would expect shows up. Since the tool goes through the tables recursively, there are some oddities worth pointing out. However, these do not get in the way of its functionality.

The first three tables outlined above are printed exactly, including the IDs (which is left out in the tables above for clarity). This can be seen in Figure 3.10.

The screenshot shows three tables displayed in a tool. The first table is titled 'user 10144' and has columns: emp\_no, birth\_date, first\_name, last\_name, gender, and hire\_date. The second table is titled 'salaries' and has columns: emp\_no, salary, from\_date, and to\_date. The third table is titled 'titles' and has columns: emp\_no, title, from\_date, and to\_date.

user 10144					
emp_no	birth_date	first_name	last_name	gender	hire_date
10144	1959-06-17	Marla	Brendel	M	1985-10-14

salaries			
emp_no	salary	from_date	to_date
10144	56720	1985-10-14	1986-10-14
10144	60264	1986-10-14	1987-10-14
10144	63968	1987-10-14	1988-10-13
10144	66772	1988-10-13	1989-10-13
10144	68160	1989-10-13	1990-10-13
10144	71280	1990-10-13	1991-10-13
10144	74279	1991-10-13	1992-10-12
10144	74378	1992-10-12	1993-08-10

titles			
emp_no	title	from_date	to_date
10144	Senior Staff	1992-10-14	1993-08-10
10144	Staff	1985-10-14	1992-10-14

FIGURE 3.10: The tool shows most tables exactly how you would expect

The last table, referred to as *Employee at departments* before, has its full contents displayed but separated in a few elements as shown in Figure 3.11.

The screenshot shows three tables displayed in a tool. The first table is titled 'dept\_emp' and has columns: emp\_no, dept\_no, from\_date, and to\_date. The second table is titled 'departments' and has columns: dept\_no, dept\_name. The third table is also titled 'departments' and has columns: dept\_no, dept\_name.

dept_emp			
emp_no	dept_no	from_date	to_date
10144	d002	1985-10-14	1988-09-02
10144	d003	1988-09-02	1993-08-10

departments	
dept_no	dept_name
d002	Finance

departments	
dept_no	dept_name
d003	Human Resources

FIGURE 3.11: The tool separates some data into different and/or multiple views as a result of the prototyping method used

Notice that the only difference is that the department names are not interpolated into the employee status data. Instead, the IDs are displayed which can then be correlated with the department names data below. An additional effect is that the data for the department names is split into two display elements, which is a side effect of the fast implementation method used to build this prototype. It could be improved in a future work.

Deletion of this data works identical to the description of how it was expected to work. All rows displayed in Figure 3.10 and Figure 3.11 are removed, except for the data about the department names.

As mentioned at the very end of Section 3.3, this is the result of using cascade deletion rules in the foreign key constraints.

### 3.4.2 Mock health sensor database

The next test will be on the database of a health sensor device. To quickly explain the device and the data it produces, the sensor measures the pressure of exhalations that a patient can produce. The data multiple exhalations is then grouped into sessions, which are groups of exhalations that closely follow each other. Those sessions belong to a specific user.

As you may notice, all data in this platform is user specific. Since the only information that gets stored is a user's profile (PII) and their exhalations, the database would be completely empty if every single user would request to be forgotten.

Since exhalation data is very vast (many measurements are taken each session), it would be impractical to show the full data here. Instead, a snippet of the data presented by the tool is shown in Figure 3.12.

user d7659a6f-3e21-4197-9518-b99f05e21342

id	first_name	last_name	birthdate	height	weight	email	password
d7659a6f-3e21-4197-9518-b99f05e21342	Martijn	Luyckx	1997-05-14	[...]	[...]	[...]	[...]

sessions

id	salbutamol	badcondition	time	user
1c322a37-467b-4d55-a3b6-21898d13839e	0	0	2020-10-02 07:07:32	d7659a6f-3e21-4197-9518-b99f05e21342
3558f0e6-b932-4267-8478-d0999cef0fd9	0	0	2020-10-25 17:02:15	d7659a6f-3e21-4197-9518-b99f05e21342
4029c795-ca0e-4619-b2c5-545b84c3d0ab	0	0	2019-08-25 05:07:32	d7659a6f-3e21-4197-9518-b99f05e21342
53a4f61a-542d-464d-9235-4c83615f9034	0	0	2019-08-30 05:05:32	d7659a6f-3e21-4197-9518-b99f05e21342
55105dfb-4c7c-41b9-aadb-0e12b0eed71e	0	0	2020-09-29 18:24:32	d7659a6f-3e21-4197-9518-b99f05e21342

exhalations

id	diameter	session	game	time
80864b43-3e1f-43f3-b201-6dc459fcd32f	35	3558f0e6-b932-4267-8478-d0999cef0fd9	Sprinter	2020-10-25 18:02:24
d22bd76e-1548-4a2b-bc59-32294884f1b1	35	3558f0e6-b932-4267-8478-d0999cef0fd9	Sprinter	2020-10-25 18:02:15

exhalations

id	diameter	session	game	time
3ced342e-da99-4475-974e-bc9222f37edd	35	4029c795-ca0e-4619-b2c5-545b84c3d0ab	Sprinter	2019-08-25 13:46:17

FIGURE 3.12: A (redacted) snippet of the information presented by the tool for Flow Connect

The same story can be told about the deletion as for the previous example: a single delete statement on the user's unique ID invokes the cascade deletion rules and erases all personal data from the database.

It can be concluded that the prototype correctly performs the tasks it was built to perform and confirms the feasibility of building a standalone tool. However, it must be noted that database design is extremely important as accessing the data relies on the foreign key constraints that were set by the developer(s). The erasing of data even relies on the deletion rules configured on those relationships.





## Chapter 4

# Informing the public after an incident

Even when software is built with utmost attention to privacy and security, incidents are bound to happen. Examples of just this year (2021) include hundreds of millions of records leaked from LinkedIn [67], a large breach leak at Allekabels, and an enormous global data leak from Facebook [30, 18].

Allekabels, a large Dutch online shop, had over 3.6 million records of private customer data leaked [1]. However, the public perception became primarily negative after people discovered that Allekabels only notified about 5000 of their customers. For example, the Dutch government authority about privacy started an investigation on the communication of Allekabels [45].

The company let its customers know that their email addresses could have been leaked [31, 32], which was an understatement. Tweakers, a Dutch technology website for reviews and news reports, stated that *"it looks like Allekabels only contacted customers about the data leak who had a unique or modified email address specifically for their account"* (originally in Dutch, translated to English).

As an additional way to improve the privacy infrastructure of existing web applications, this chapter will explore ways to mitigate the reputation damage of those inevitable privacy incidents. It will compare companies that communicated transparently against those which did not. The chapter will also present a case study for which a tool was built to give victims a means of finding out the impact to them personally and what to do about it.

### 4.1 Types of privacy-related incidents

As this chapter talks extensively about privacy incidents, specifically data breaches, it is important to highlight some key ways such an incident can occur and explain some nuances of different jargon.

Generally, a data breach or leak occurs after one of the following scenarios:

- **Hacking** —A malicious hacker exploits a technical flaw in a digital system (sometimes by injecting a payload) to extract sensitive data.
- **Unauthorised disclosure** —Unethical actors create a large, coherent database of information that can be found or interpolated from publicly available information on a digital system (albeit unintentionally).

- **Theft** —Users of a digital system or insiders who have elevated access to that digital system are targeted by social engineering (most commonly phishing) to steal their credentials and extract their sensitive information.

There are other scenarios, like improper data disposal, which causes the data to get into the wrong hands, but these are rather uncommon, as shown in Figure 4.1[5].

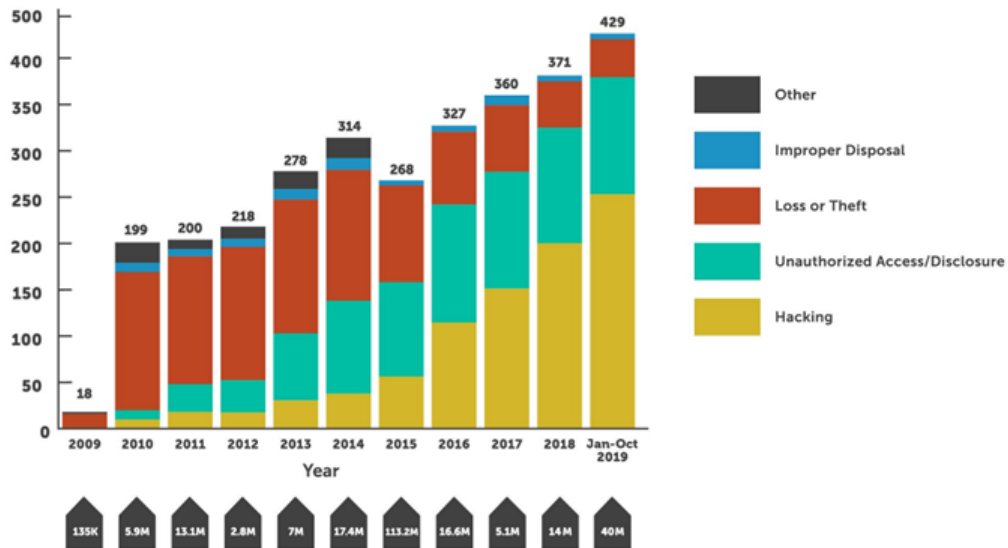


FIGURE 4.1: Graph depicting yearly occurrences of the kinds of data breaches by Enisa [5]

### 4.1.1 Hacking

*"The essence of hacking is finding unintended or overlooked uses for the laws and properties of a given situation and then applying them in new and inventive ways to solve a problem-whatever it may be",* is in the introduction of Jon Erickson's book *"Hacking, the art of exploitation"* [22].

In these increasingly digital times, it is impossible to guarantee to write bug-free code. One may not be able to predict every way in which multiple parts of a system could interact with each other, may not understand lower-level systems they are building on top of, not know about certain attacks or just make a simple human error like a typo.

All of that is to say that a malicious user will most likely find some way to use a digital system in an unintended way to bypass some protections or extract sensitive data [5]. The impact of such a hack will vary based on the type of attack and the circumstances, but it is best to be aware that this could happen and try to mitigate this as much as possible.

There are two main ways to defend against hacking:

- Defensive mitigation
- Offensive diagnosis

Defensive mitigation for hacking starts when building the application. The programmer can use secure coding practices [51], limit the use of third party code, which could introduce vulnerabilities without being able to control them or use the principle of least privilege [52], to name a few examples.

You can also offensively try to improve the security of an application by thinking like the malicious hacker: try to attack your system and see in which ways you succeed. This is widely known as *ethical hacking*. Then, the programmers can fix the issues that were found by the ethical hacker (preferably solving the root cause and not just patching the symptom). Traditionally, there are two ways of approaching this:

- Penetration testing (fixed time span)
- Crowd-sourced testing (continuous)

When a company wants to harden its security in an offensive way, the former option consists of hiring one or more ethical hackers. This is called penetration testing [64]. The hackers will test the application's security, usually in a fixed amount of time, by finding vulnerabilities in the system. Afterwards, they summarise their findings in a report to communicate their work to the company. This way, the company has a snapshot of their security.

Alternatively, a company can opt to tap into the knowledge of the crowd. The concept of setting up a responsible disclosure program or bug bounty program is that everyone (or a select group of invited people) can report vulnerabilities found in the system to the company [76]. Optionally, the company can reward those findings (e.g. by issuing a monetary reward) to incentivise ethical handling of those findings.

Naturally, all these methods can be combined to offer a defensive and dynamically offensive mitigation strategy against hacking.

#### 4.1.2 Unauthorised disclosure

Scraping is a popular form of unauthorised disclosure of information. In essence, a malicious actor will (robotically) iterate over all public information on a digital system, usually personal(ly identifiable) information, like name, email address, approximate location or more. This way, they create a database of information that can be very valuable to social engineers or advertisers. Occasionally, the line between scraping and hacking can be thin, like when a hacker used an alleged misconfiguration / unfortunate default of Facebook's *who can look me up* feature (which is more thoroughly explained in the upcoming Section 4.2) to iterate over many user's phone number.

#### 4.1.3 Theft

An attacker can steal information by tricking a victim into giving it to them (usually unknowingly), which is called phishing. A recent study, *An In-Depth Benchmarking and Evaluation of Phishing Detection Research for Security Needs*, defines phishing as "a popular form of social engineering attack wherein the attacker deceives a victim through impersonation" [20]. Common attack vectors include emails, websites or social media posts/messages which contain a malicious link that tricks the victim into compromising their system or entering confidential information like credentials.

The compromised system of account can then be used to steal information from the victim. These attacks are most commonly made on a large scale as opposed to targeted attacks (called spear phishing [6]).

## 4.2 Case study: Facebook's global data leak in 2021

In April 2021, an anonymous hacker leaked 533,313,128 records of Facebook users' data for free [2]. Most notably, the leaked dataset matched those many millions of Facebook users' phone numbers to their Facebook ID (and thus their name and profile picture). For many records, the data is supplemented with publicly available data from each profile, including gender, location, relationship status, occupation, date of birth, and email addresses.

### 4.2.1 Timeline and communication

In 2017, Belgian ethical hacker Inti De Ceukelaire reported a scraping vulnerability [60] to Facebook with which it was possible to obtain an arbitrary Facebook user's phone number. As an example, he proved that he was able to get the number of Jan Jambon, a high profile Belgian politician [43]. This was caused by the privacy feature *who can look me up* being set to *everyone*, which was the default setting for all users. Facebook did not consider this a vulnerability and did not plan to make any changes (except that it *may* tweak some rate limits) based on the report by De Ceukelaire [42] (as shown in Figure 4.2).



FIGURE 4.2: Facebook's response to the report by Inti De Ceukelaire [8]

Two years later, in January 2021, a sale of the phone numbers was announced. Using an automated Telegram bot, the anonymous hacker sold look-ups for \$20 each (with bulk sales as low as \$0.50 per lookup if an actor was willing to spend \$5,000 total) [15]. In Vice's article about the bot, Alon Gal stated that *"it is important that Facebook notify its users of this breach, so they are less likely to fall victim to different hacking and social engineering attempts"* [15]. Unfortunately, Facebook did not respond.

At last, three more months after that, the inevitable happened. In April 2021, all of the leaked records were published for free on a hacker forum. With that, over half a billion people had their phone numbers and identity information exposed to everyone with rudimentary data skills.

As early as April 3rd, 2021, news outlets were reporting about the Facebook data leak [25, 30, 41]. Alon Gal, the author of the security blog *"Under The Breach"* and

author of a popular Twitter thread about the Facebook leak, said that they *"have yet to see Facebook acknowledging this absolute negligence of your data"*.

A few days pass, but the only statement that Facebook gave the press is that the data is "old", "already public" [58], and that they already fixed the issue which made scraping possible in 2019 [30]. Apart from that, the only communication that Facebook put out in the world, albeit unintentionally, was a leaked internal email from April 8[4]. In there, it is clear that Facebook's communication strategy favours their reputation over the privacy of their users. In the email, they communicate in a way to deflect blame away from themselves.

A section in the section about their long-term strategy sums it up nicely: *"We expect more scraping incidents and think it's important to both frame this as a broad industry issue and normalise the fact that this activity happens regularly"*.

## 4.2.2 Compromised credential checking

In the few days after the news about the free publication of the leaked records, many look-up tools spawned in the wild. One of them is developed by the author of this work in the context of this thesis, and the subject of this section: **Ben Ik Erbij?** [46] (which is Dutch for *Am I Included?*). All the tools had the same purpose, which is letting everyone (or in our case, people located in Belgium or The Netherlands) check for themselves if their phone number was included in this leak and sometimes give privacy-related advice. Other tools include *Facebook Checker* [63] and *Is Mijn Nummer Gelekt?* [50] (which is Dutch for *Is my number leaked?*).

### Reasoning

For every major breach or leak, people want to know what it means for them personally. That's why on average, 150,000 people check if their email address(es) were included in any breach indexed by *Have I Been Pwned?* every day (with spikes in the millions of unique visitors in a day) [33].

Unfortunately, *Have I been Pwned?* could only perform a search on an email address (which was about to change due to this Facebook Leak [34]). That means that it is unsuitable for informing the public about this leak, as its main source of information was a phone number and Facebook ID (see Section 4.2). Only about 2.5 million records contained an email address, which is roughly 0,45% of all leaked records. Clearly not enough to get any kind of reassurance that your data was not leaked if *Have I Been Pwned* did not find your email address in the Facebook leak.

### Method of searching

To be reliable, those look-up tools need to search on the main source of data. Either phone numbers or Facebook IDs/profiles would work. Tools were created for both methods of searching, each with its own upsides and downsides.

For one, it could be argued that implementing an input field in a privacy-related tool where users are required to enter their phone number is a bad practice. It would potentially help to normalise entering your phone number anywhere. Additionally, a bad actor could use this method to link IP addresses with phone numbers. Lastly, logging every number in there could be a good way to gather phone numbers that



are likely in active use (as users would be expected to enter the phone number they actively use).

The counterargument here is that the bad actor would easily have access to the full list of leaked information anyway, eliminating the benefits of logging phone numbers.

The familiarity with phone numbers and their importance are the main arguments for using those as the main method of searching. Additionally, by searching by phone number, users could get a more hands-on feel for the fact that their phone number had become public information. They do not need to make the mental connection where they enter their Facebook URL link (which may feel like public information already), see that they are included in the leak, and then realise that it means that their phone number is now public information.

Under the assumption that using a phone number as the method of searching would be a lower barrier to use the tool, and thus provide more representative usage statistics, Ben ik erbij? opted to use that. The following paragraphs will explain the technical details of the lookup tool, how it could have been built to actively anonymise interaction with the tool and how it was perceived by the public to conclude if it is beneficial to use this to aid your communication to the public in the event of a data leak.

### Technical implementation

As a short summary to start with, the tool was straightforward under the hood as it had to be built from the ground up in a few hours to increase the odds of getting picked up by the public. That also means there is room to discuss how it would have been built in an ideal scenario.

Users were prompted to enter their phone number on the page (the page helped them get the formatting human-friendly but consistent: +32 4XX XX XX for Belgium or +31 6XX XX XX for The Netherlands), which was then submitted with a POST request in a standardised format (324XXXXXXXX [BE] or 316XXXXXXXX [NL]). The server then queried its own Postgres database, which contained a single table with a single column. That column was a BIGINT type so that an optimised index could be created to speed up lookups:

```
CREATE INDEX "phone-index"
  ON numbers USING btree
  (phone DESC NULLS LAST)
;

ALTER TABLE numbers
  CLUSTER ON "phone-index";
```

The tool queries for phone numbers quickly and easily with prepared statements like so:

```
SELECT phone
FROM numbers
WHERE phone = ?
LIMIT 1;
```



The `LIMIT 1` ensures that the database will stop as soon as it found a match, decreasing response times. The web server and database were deployed on a Google Cloud Engine using Apache, PHP, and, as already mentioned, Postgres. Because of occasional slowness during peak times, this work did experimental research on which technology stack would have performed the fastest. Three combinations were tested:

- PHP (Apache) + Postgres
- NodeJS + Postgres
- NodeJS + Redis

The technologies were chosen for specific characteristics. NodeJS was picked because it could open a single connection to the database and reuse that for every query it needed to do. This is in contrast to PHP, which spawns a new process for every request and has to re-establish the database connection every time. The hypothesis is that NodeJS would perform better for this reason. As the alternative database technology, Redis was chosen. Since it is an in-memory key-value lookup system, it is expected to work faster than Postgres, which operates relationally. A python script was used to benchmark the different options (all running on the same Google Cloud Engine instance):

```

LOOPS = 5000
TEST_SUBJECTS = [
    ["NodeJS + Redis", "http://localhost:8080/api", 0],
    ["Apache PHP + Postgres", "http://localhost:80/postgres-api.php", 0],
    ["NodeJS + Posgres:", "http://localhost:8081/api", 0],
]

import requests
import time
import random

maxloops = 500000 # size of random_numbers.csv
numbers = []
count = 0
with open('random_numbers.csv', 'r') as numbers_file:
    while count < LOOPS and count < maxloops:
        numbers.append(numbers_file.readline()[:11])
        count += 1

random.shuffle(numbers)

print("Starting benchmark of %d random numbers\n" % len(numbers))

for number in numbers:
    for subject_nr in range(len(TEST_SUBJECTS)):
        subject = TEST_SUBJECTS[subject_nr]
        start = time.time_ns()
        x = requests.post(subject[1], data = number)
        stop = time.time_ns()

```

```
TEST_SUBJECTS[subject_nr][2] += stop-start

for subject in TEST_SUBJECTS:
    print("%s: %d ns" % (subject[0], subject[2]))
    print("That is %f seconds\n" % (subject[2]/1e9))
    print()
```

To conduct the test, `random_numbers.py` was generated. It consisted of 250,000 random numbers from the breach and 250,000 randomly generated phone numbers (which were expected to not be present in the leaks database, but not guaranteed to not be in there). This way, a randomly selected phone number from the file had about a 50% chance of being in the leaked data set.

The results confirmed the hypothesis, as can be seen in Figure 4.3. The NodeJS + Redis combo was 72% faster than the deployed Apache-PHP + Postgres method. Interestingly, the switch from Postgres to Redis only accounted for 6% of that improvement; the other 66% can be credited to NodeJS and its ability to sustain the database connection.

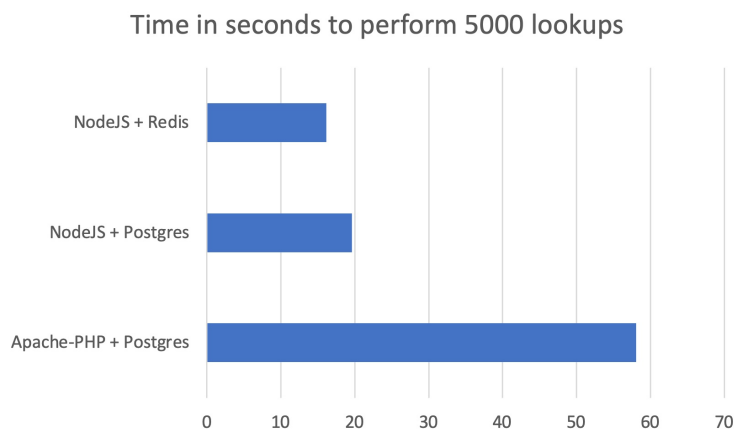


FIGURE 4.3: Results of the benchmark on different technology stacks to perform phone number lookups in a database

Another potential improvement is the aforementioned active anonymisation of data. In theory, an attacker with a man in the middle position [49] (which is a realistic scenario [66]) could collect any data which was sent over a network connection. In the case of Ben Ik Erbij?, that would be phone numbers that are expected to be mostly valid and in active use.

To prevent that, a similar concept to k-anonymity [65] could be used: make sure that the input for the service is one-way encoded (hashed) [55] and make sure that for every input the user gives, there are multiple outputs per definition. Then, return the set of those outputs and make the user check for themselves if any of the suffixes match theirs. That way, the difficulty for obtaining the users' information increases dramatically for the eavesdropping bad actor [44]. This is depicted schematically in Figure 4.4.

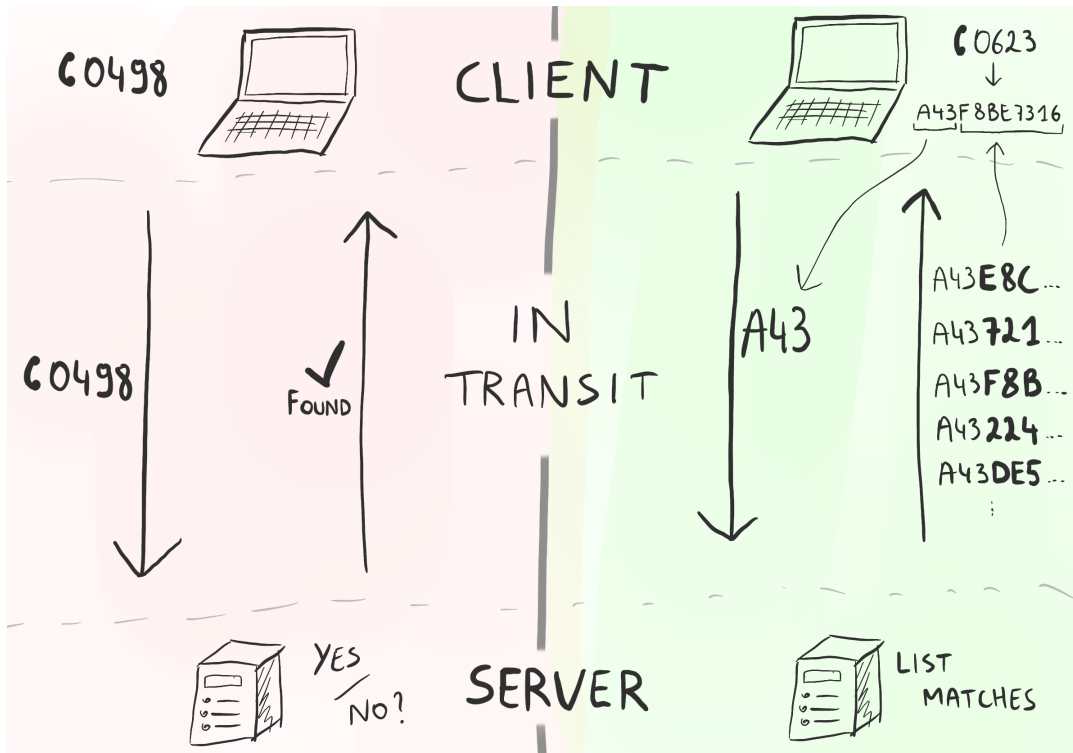


FIGURE 4.4: Left: unaltered network communication of Ben Ik Erbij?  
- Right: Schematic depiction of a data anonymisation method

The following paragraphs will go over a realistic implementation of such a system. Let's start with the numbers: in total, there are 8,592,995 phone numbers in the database. 3,164,351 of those are from Belgium numbers (starting with 324) and 5,428,644 of those are from The Netherlands (starting with 326). For this example, SHA1 was picked (40 characters of base 16). If it is desired that on average 10 results are found for any given prefix, it can be calculated how long that prefix should be for this database of 8.5 million records. To start, solve this equation:

$$values = \frac{8,592,995}{10} = 859,299.5$$

From that, it follows that a prefix length needs to be chosen that can represent 429,650 unique values. Since the hash is base 16, we can calculate the length of the prefix as follows:

$$length_{prefix} = \frac{\ln(859,299.5)}{\ln(16)} = 4.9282004$$

Obviously, the length of the prefix can only be a whole number. So the closest options are a prefix that is 4 characters long or one that is 5 characters long. Let's calculate the matches these would give on average:

$$avg\_length_{prefix-4} = \frac{8,592,995}{16^4} \approx 131$$

$$avg\_length_{prefix=5} = \frac{8,592,995}{16^5} \approx 8$$

It is unnecessary to declare the optimal prefix length in this work. A shorter length would result in more thorough anonymisation (131 results on average) but consume more bandwidth for each look-up. The difference in bandwidth is not very large (especially considering the current infrastructure), but it could definitely be significant in a large scale tool. As a SHA1 hash is 20 bytes in size, the server reply for 8 hashes would be  $8 * 20bytes = 160bytes$ , where the reply for 131 hashes would be  $131 * 20bytes = 2,620bytes = 2.62kilobytes$ .

About two and a half kilobytes of data per lookup (not compressed) seems reasonable considering that the page sent to the user to show the form (HTML, CSS, Javascript) is most likely considerably larger than that. For example, the very minimalistic page of *Ben Ik Erbij?* which does not use any dependencies or extra files, weighs 4.0 kilobytes. If this kind of anonymisation would have been implemented in *Ben Ik Erbij?*, the chosen prefix length would most likely have been 4 to prioritise privacy over bandwidth.

## Results

The tool *Ben Ik Erbij?* was released in a few different layouts and on a few domains at first, but ultimately settled on providing a unified experience for all people who wanted to check a Belgian or Dutch number (albeit served in a localised environment). The final version can be seen in Figure 4.5. To increase the likelihood of being picked up by the public, Inti De Ceukelaire assisted in informing the public about the tool, as he has established contacts in Belgian and Dutch media, as well as a sizeable following on social media [9, 10].

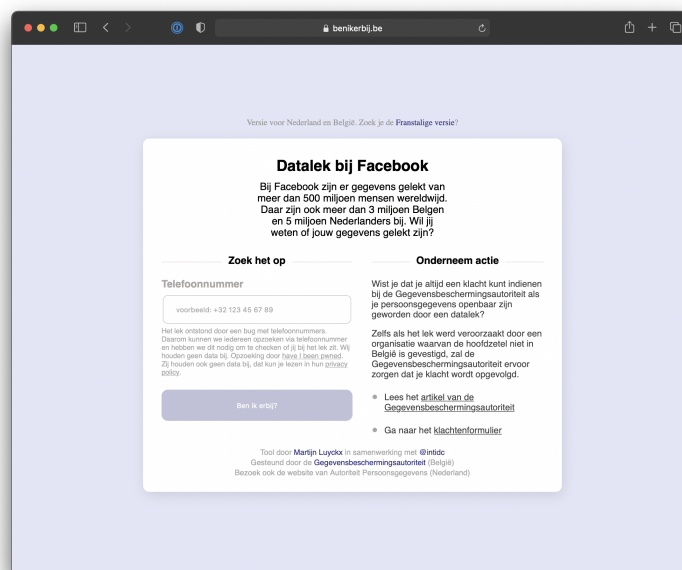


FIGURE 4.5: A screenshot of the tool 'Ben Ik Erbij?'

Local and national news outlets picked up *Ben Ik Erbij?* as the de facto tool to check whether or not your personal phone number had become public information and what to do about it [54, 53, 68, 56]. This fact alone is a good indicator that the public is very interested in (personalised) information about a security breach. However, the statistics of *Ben Ik Erbij?* show just how much interest there really was, as depicted in Figure 4.6 and Figure 4.7.

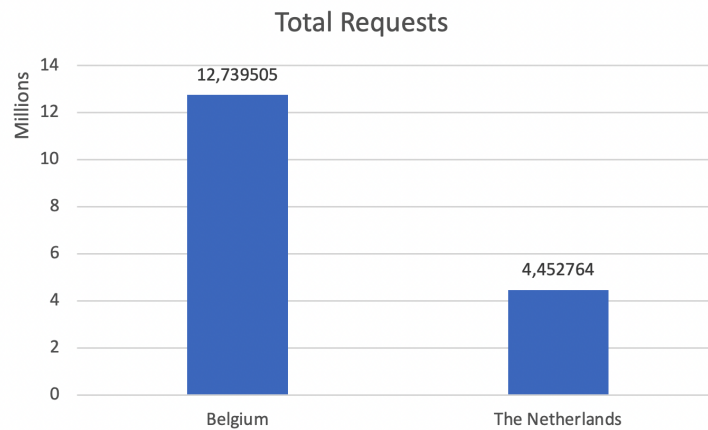


FIGURE 4.6: Total requests to *Ben Ik Erbij?* per country based on domain TLDs as counted by Cloudflare

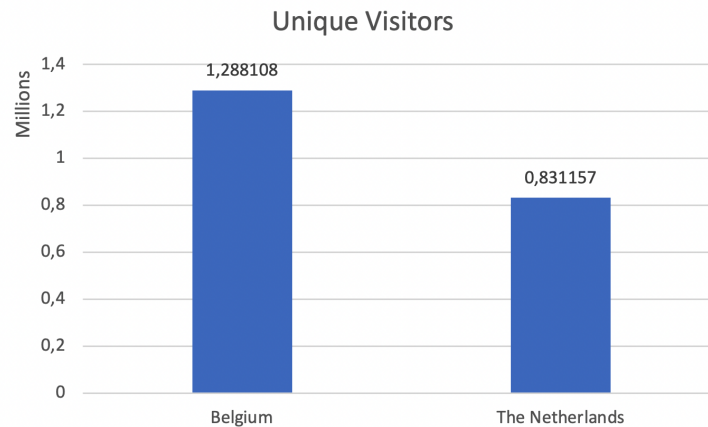


FIGURE 4.7: Unique visitors to *Ben Ik Erbij?* per country based on domain TLDs as counted by Cloudflare

Interestingly, it was very visible on the CPU usage charts when the *Ben Ik Erbij?* was mentioned on a live news reporting (TV, radio...) as can be seen in Figure 4.8.

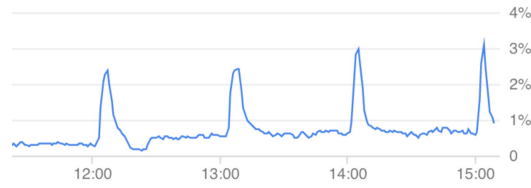


FIGURE 4.8: CPU usage peaks of *Ben Ik Erbij?*'s server corresponding to live news reports based on Google Cloud's analytics page

Using these statistics, it is also possible to calculate an interval of how many actual look-ups have been done with the tool. The tool did not log anything itself, so it is impossible to get the actual number. Starting with Belgium, 1.29 million unique visitors loaded the tool. That means that at least 1.29 million requests of the total requests were page loads (and thus not look-ups).

$$look - ups_{maximum} = 12.74million - 1.29million = 11.45million$$

That leaves 11.18 million look-ups as the maximum for Belgium. More realistically, visitors came back later to do another look-up (for example, for a friend or relative) and had to load the page again. Assuming the number of users who loaded the page but did not perform a look-up is rather low, we can calculate a realistic minimum amount of look-ups by reasoning that every lookup is preceded by a page load.

$$look - ups_{minimum} = \frac{12.74million}{2} = 6.37million$$

This means that the number of look-ups done will most likely fall in the interval [6,370,000, 11,450,000] for Belgians (where the tool was most popular). Considering the population of Belgium is 11,646,084 at the time of writing [74], of which 6.9 million actively use Facebook [40], it can be concluded that there is a high interest in the service that *Ben Ik Erbij?* provided.

Similarly, the interval for The Netherlands can be calculated:

$$look - ups_{maximum} = 4.45million - 0.82million = 3,62million$$

$$look - ups_{minimum} = \frac{4.45million}{2} = 2.23million$$

Given that the tool was significantly less reported in The Netherlands, about 3 million look-ups is still very significant with a population of 17,177,146 people at the time of writing [75], of which 10.4 million people actively use Facebook (data from 2020) [29].

### Belgian Data Protection Authority

On April 7, the Belgian Data Protection Authority (formerly known as the Belgian Privacy Commission) had reached out to *Ben Ik Erbij?* via email to discuss a potential collaboration. The Belgian Data Protection Authority wanted to achieve two things:

- Collect a large number of Belgian complaints about this Facebook data leak.
- Prevent additional data leaks by communicating officially about *Ben Ik Erbij?*. This way, the chance of potential victims entering their data on malicious tools is reduced.

A few hours after the email, the Belgian Data Protection Authority and *Ben Ik Erbij?* discussed the details of the collaboration thoroughly and concluded it would be beneficial to expand the reach of the tool to Wallonia, the French-speaking part of Belgium, by building a French version. That did not exist up until that point. The Belgian Data Protection Authority offered its help to translate the page, of which the result can be seen on [jesuisconcerne.be](https://jesuisconcerne.be).

As part of the collaboration, *Ben Ik Erbij?* informed the visitors of the tool about the Belgian Data Protection Authority and its quest to collect formal complaints from Belgian citizens. Ultimately, the Belgian Data Protection Authority could not disclose the number of complaints they received. David Stevens, chairman of the organisation, was able to state that they "*received a very significant amount of complaints as a result of the collaboration with Ben Ik Erbij?*". That is another reason from which it can be concluded that the public has a great need to be (personally) informed about data leaks like this, and a tool like *Ben Ik Erbij?* succeeds in doing that.

## Chapter 5

# Conclusion

Privacy of personal data is more important than ever, definitely since many people's personal data is at least online \*somewhere\* nowadays. From social media to forums, accounts with extensive data on people are prevalent. Securing that data is crucial to privacy and helps prevent malicious hackers from succeeding in exploiting victims.

Regulators are stepping up (GDPR, CCPA...), which is definitely an excellent thing and a big step in the right direction. Unfortunately, not every web application complies with the regulations that are now in place. It is not evident to build an application that is fully compliant with regulations, and it is definitely not easy to update existing web applications to better secure personal data.

This thesis provides a working prototype that enables many existing web applications to comply with a part of those regulations without altering their codebase, proving its feasibility. By merely connecting the tool to their database and following an intuitive setup process, they can handle GDPR rights requests semi-automatically. This lowers the barrier to increase compliance for existing web applications.

Additionally to the aforementioned benefits, the tool also assists all parties concerning GDPR's seven data processing principles. Specifically, *Lawfulness, fairness and transparency, Accuracy, Storage limitation, and Accountability* are supported as explained in the Chapter 3's introduction.

When a security incident happens, and a data leak occurs, which is inevitable even for the largest applications on the web, it is important to transparently inform your customers about it and let them know how it affects them. In the case study around the data leak of Facebook in April 2021, the effectiveness of a tool that does just that was analysed. The public interest in the tool was very high. National news outlets organically started to report on the tool frequently. The Belgian national instance on data protection (DPA) asked to collaborate. Belgian citizens did over 6 million look-ups for personal information, a country with 11.5 million inhabitants.

Of course, more work can be done on the topic. A user study can be conducted on the tool for semi-automatic handling of GDPR rights requests, researching if its functionality is understood and appreciated in real-world situations. Additionally, it could be expanded to more technology stacks or implemented in a major CMS system (as a plugin, for example).





# Bibliography

- [1] Test Aankoop. *Gegevens van 3,6 miljoen klanten gestolen bij Allekabels.nl. Ook Belgen mogelijk getroffen*. URL: <https://www.test-aankoop.be/familie-privé/consumentenrechten/nieuws/datalek-allekabels> (visited on 07/02/2021).
- [2] Lawrence Abrams. *533 million Facebook users' phone numbers leaked on hacker forum*. URL: <https://www.bleepingcomputer.com/news/security/533-million-facebook-users-phone-numbers-leaked-on-hacker-forum/> (visited on 07/15/2021).
- [3] FileMaker Pro 18 Advanced. *One-to-many relationships*. URL: [https://fmhelp.filemaker.com/help/18/fmp/en/index.html#page/FMP\\_Help/one-to-many-relationships.html](https://fmhelp.filemaker.com/help/18/fmp/en/index.html#page/FMP_Help/one-to-many-relationships.html) (visited on 08/14/2021).
- [4] BCC. *Facebook downplays data breach in internal email*. URL: <https://www.bbc.com/news/technology-56815478> (visited on 08/05/2021).
- [5] ENISA Threat Landscape 2020 Data Breach. *Data breach - ENISA Threat Landscape*. URL: <https://www.enisa.europa.eu/publications/enisa-threat-landscape-2020-data-breach> (visited on 08/12/2021).
- [6] Deanna D Caputo et al. "Going spear phishing: Exploring embedded training and awareness". In: *IEEE security & privacy* 12.1 (2014), pp. 28–38.
- [7] Ann Cavoukian. *Embed Privacy by Design and Reap the Rewards; Privacy AND Data Utility!* URL: [https://isotc.iso.org/livelink/livelink/19749828/05\\_-\\_Ann\\_Cavoukian\\_-\\_ISO\\_COPOLCO\\_PLENARY\\_BALI\\_2018.mp4?func=doc.Fetch&nodeid=19749828](https://isotc.iso.org/livelink/livelink/19749828/05_-_Ann_Cavoukian_-_ISO_COPOLCO_PLENARY_BALI_2018.mp4?func=doc.Fetch&nodeid=19749828) (visited on 11/16/2020).
- [8] Inti De Ceukelaire. *Facebook: alle GSM nrs zijn standaard openbaar. Kan het enkel manueel op 'enkel vrienden' zetten. Duurt maar 30 min om nr te achterhalen...* URL: <https://twitter.com/intidc/status/819836039057514496> (visited on 08/01/2021).
- [9] Inti De Ceukelaire. *Twitter account @intidc*. URL: <https://twitter.com/intidc> (visited on 08/12/2021).
- [10] Inti De Ceukelaire. *Twitter account @securinti*. URL: <https://twitter.com/securinti> (visited on 08/12/2021).
- [11] Edgar F Codd. "A relational model of data for large shared data banks". In: *Software pioneers*. Springer, 2002, pp. 263–294.
- [12] ISO: COPOLCO. *WORKSHOP ON CONSUMER PROTECTION IN THE DIGITAL ECONOMY*. URL: [https://isotc.iso.org/livelink/livelink/fetch/-8925011/8925029/8925038/17692204/19549786/11\\_-\\_Meeting\\_report\\_of\\_the\\_COPOLCO\\_workshop\\_-\\_Consumer\\_protection\\_in\\_the\\_digital\\_economy.docx.pdf?nodeid=19769477&vernum=-2](https://isotc.iso.org/livelink/livelink/fetch/-8925011/8925029/8925038/17692204/19549786/11_-_Meeting_report_of_the_COPOLCO_workshop_-_Consumer_protection_in_the_digital_economy.docx.pdf?nodeid=19769477&vernum=-2) (visited on 11/16/2020).
- [13] Oracle Corporation. *13.1.17.5 FOREIGN KEY Constraints*. URL: <https://dev.mysql.com/doc/refman/5.6/en/create-table-foreign-keys.html> (visited on 08/15/2021).

- [14] Oracle Corporation. *Employees Sample Database*. URL: <https://dev.mysql.com/doc/employee/en/> (visited on 08/15/2021).
- [15] Joseph Cox. *Bot Lets Hackers Easily Look Up Facebook Users' Phone Numbers*. URL: <https://www.vice.com/en/article/xgz7bd/facebook-phone-numbers-bot-telegram> (visited on 08/01/2021).
- [16] *Database Systems, The Complete Book, 2nd edition*. Pearson, 2013. ISBN: 9781292024479. URL: <https://www.pearson.com/uk/educators/higher-education-educators/program/Ullman-Database-Systems-Pearson-New-International-Edition-The-Complete-Book-2nd-Edition/PGM1048677.html>.
- [17] Dataedo. *Default databases in MySQL*. URL: <https://dataedo.com/kb/databases/mysql/default-databases-schemas> (visited on 08/15/2021).
- [18] DataNews. *Gegevens van half miljard Facebook-gebruikers gelekt, daarvan drie miljoen uit België*. URL: <https://datanews.knack.be/ict/nieuws/gegevens-van-half-miljard-facebook-gebruikers-gelekt-daarvan-drie-miljoen-uit-belgie/article-news-1719287.html> (visited on 07/02/2021).
- [19] Datanyze. *Databases Market Share Report*. 2020. URL: <https://www.datanyze.com/market-share/databases--272> (visited on 11/02/2020).
- [20] Ayman El Aassal et al. "An In-Depth Benchmarking and Evaluation of Phishing Detection Research for Security Needs". In: *IEEE Access* 8 (2020), pp. 22170–22192. DOI: 10.1109/ACCESS.2020.2969780.
- [21] Equality and Human Rights Commission. *Human Rights Act: Article 8: Respect for your private and family life*. URL: <https://www.equalityhumanrights.com/en/human-rights-act/article-8-respect-your-private-and-family-life> (visited on 11/17/2020).
- [22] Jon Erickson. *HACKING - THE ART OF EXPLOITATION*. No Starch Press, 2008. URL: [https://books.google.be/books?id=0FW3DMNhl1EC&dq=what+is+hacking&lr=&source=gbs\\_navlinks\\_s](https://books.google.be/books?id=0FW3DMNhl1EC&dq=what+is+hacking&lr=&source=gbs_navlinks_s).
- [23] Facebook. *How do I download a copy of my information on Facebook?* URL: <https://www.facebook.com/help/212802592074644> (visited on 08/23/2021).
- [24] Artem Minaev from First Site Guide. *CMS Market Share 2021: Usage Statistics of Popular Technologies*. URL: [CMSMarketShare2021:UsageStatisticsofPopularTechnologies](https://www.firstsiteguide.com/cms-market-share-2021-usage-statistics-of-popular-technologies/) (visited on 08/13/2021).
- [25] Alon Gal. *All 533,000,000 Facebook records were just leaked for free*. URL: <https://twitter.com/UnderTheBreach/status/1378314424239460352> (visited on 08/03/2021).
- [26] GDPR.EU. *What are the GDPR Fines?* 2018. URL: <https://gdpr.eu/fines/> (visited on 11/17/2020).
- [27] GDPR.EU. *What is GDPR, the EU's new data protection law?* 2018. URL: <https://gdpr.eu/what-is-gdpr/> (visited on 11/17/2020).
- [28] Prisma's Data Guide. *Comparing database types: how database types evolved to meet different needs*. URL: <https://www.prisma.io/dataguide/intro/comparing-database-types> (visited on 08/14/2021).
- [29] drs. H. Hoekstra. *Social media onderzoek 2020*. URL: <https://www.newcom.nl/social-media-2020/> (visited on 08/12/2021).

- [30] Aaron Holmes. *533 million Facebook users' phone numbers and personal data have been leaked online*. URL: <https://www.businessinsider.com/stolen-data-of-533-million-facebook-users-leaked-online-2021-4> (visited on 07/02/2021).
- [31] Julian Huijbregts. *Allekabels waarschuwt 5000 klanten voor datalek van onbekende gegevens*. URL: <https://tweakers.net/nieuws/177704/allekabels-waarschuwt-5000-klanten-voor-datalek-van-onbekende-gegevens.html> (visited on 07/02/2021).
- [32] Julian Huijbregts. *Datalek bij Allekabels betrof niet 5000 maar 3,6 miljoen mensen*. URL: <https://tweakers.net/nieuws/180538/datalek-bij-allekabels-betrof-niet-5000-maar-3-komma-6-miljoen-mensen.html> (visited on 07/02/2021).
- [33] Troy Hunt. *Project Svalbard: The Future of Have I Been Pwned*. URL: <https://www.troyhunt.com/project-svalbard-the-future-of-have-i-been-pwned/> (visited on 08/05/2021).
- [34] Troy Hunt. *The Facebook Phone Numbers Are Now Searchable in Have I Been Pwned*. URL: <https://www.troyhunt.com/the-facebook-phone-numbers-are-now-searchable-in-have-i-been-pwned/> (visited on 08/06/2021).
- [35] Peter Hustinx. "Privacy by design: delivering the promises". In: *Identity in the Information Society* 3.2 (2010), pp. 253–255.
- [36] Cookie Information. *WP GDPR Compliance*. URL: <https://wordpress.org/plugins/wp-gdpr-compliance/> (visited on 08/13/2021).
- [37] Privacy International. *Part 3: Data Protection Principles*. URL: <https://dev.mysql.com/doc/employee/en/> (visited on 08/19/2021).
- [38] ISO.org. *DATA PRIVACY BY DESIGN: A NEW STANDARD ENSURES CONSUMER PRIVACY AT EVERY STEP*. URL: <https://www.iso.org/news/ref2291.html> (visited on 11/16/2020).
- [39] ISO.org. *ISO AND CONSUMERS: COPOLCO*. URL: <https://www.iso.org/copolco.html> (visited on 11/16/2020).
- [40] Simon Kemp. *DIGITAL 2021: GLOBAL OVERVIEW REPORT*. URL: <https://datareportal.com/reports/digital-2021-global-overview-report> (visited on 08/12/2021).
- [41] Jurgita Lapienytė. *Facebook data leak – 533M Facebook users' personal data leaked online*. URL: <https://cybernews.com/news/leaker-says-they-are-offering-private-details-of-500-million-facebook-users/> (visited on 08/03/2021).
- [42] Pieterjan Van Leemputten. *Facebook wil telefoonnummerlek niet oplossen*. URL: <https://datanews.knack.be/ict/nieuws/facebook-wil-telefoonnummerlek-niet-oplossen/article-normal-801399.html> (visited on 08/01/2021).
- [43] Pieterjan Van Leemputten. *Facebooks gigantische datalek werd veroorzaakt door... Facebook*. URL: <https://datanews.knack.be/ict/nieuws/facebook-s-gigantische-datalek-werd-veroorzaakt-door-facebook/article-longread-1720065.html> (visited on 07/30/2021).
- [44] Lucy Li et al. "Protocols for Checking Compromised Credentials". In: Nov. 2019, pp. 1387–1403. ISBN: 978-1-4503-6747-9. DOI: 10.1145/3319535.3354229.

- [45] Cédric Van Loon. *Enorm datalek bij Allekabels met 3,6 miljoen persoonsgegevens*. URL: <https://itdaily.be/nieuws/security/allekabels-datalek-persoonsgegevens/> (visited on 08/06/2021).
- [46] Martijn Luyckx. *Datalek bij Facebook*. URL: <https://benikerbij.be> (visited on 08/05/2021).
- [47] Justin Macmullan. *Key Consumer Issues in the digital economy and the current landscape*. URL: [https://isotc.iso.org/livelink/livelink/fetch/-8925011/8925029/8925038/17692204/19549786/01b\\_-\\_Justin\\_McMullan\\_-\\_Script\\_9\\_May\\_2018.pdf?nodeid=19752328&vernum=-2](https://isotc.iso.org/livelink/livelink/fetch/-8925011/8925029/8925038/17692204/19549786/01b_-_Justin_McMullan_-_Script_9_May_2018.pdf?nodeid=19752328&vernum=-2) (visited on 11/16/2020).
- [48] Justin Macmullan. *THE ROLE OF STANDARDS IN ADDRESSING KEY CONSUMER ISSUES IN THE DIGITAL ERA*. 2018. URL: [https://isotc.iso.org/livelink/livelink/fetch/-8925011/8925029/8925038/17692204/19549786/01a\\_-\\_Key\\_consumer\\_issues\\_in\\_the\\_digital\\_economy\\_and\\_the\\_current\\_landscape\\_by\\_Mr\\_Justin\\_MacMullan\\_Consumers\\_International.pdf?nodeid=19714876&vernum=-2](https://isotc.iso.org/livelink/livelink/fetch/-8925011/8925029/8925038/17692204/19549786/01a_-_Key_consumer_issues_in_the_digital_economy_and_the_current_landscape_by_Mr_Justin_MacMullan_Consumers_International.pdf?nodeid=19714876&vernum=-2) (visited on 11/16/2020).
- [49] Avijit Mallik. *MAN-IN-THE-MIDDLE-ATTACK: UNDERSTANDING IN SIMPLE WORDS*. URL: <https://jurnal.ar-raniry.ac.id/index.php/cyberspace/article/view/3453/2707> (visited on 08/07/2021).
- [50] Mariano Di Martino. *Is mijn nummer gelekt?* URL: <https://www.marianodimartino.com/ismijnnummergelekt/index.php> (visited on 08/05/2021).
- [51] Khrístos Mathás, Mathas Minás, and Min Christos. *Secure coding practices for web applications*. URL: <https://dione.lib.unipi.gr/xmlui/handle/unipi/13586> (visited on 08/12/2021).
- [52] Travis McPeak. “Least Privilege: Security Gain without Developer Pain”. In: *Enigma 2018 (Enigma 2018)*. Santa Clara, CA: USENIX Association, Jan. 2018. URL: <https://www.usenix.org/node/208140>.
- [53] De Morgen. *Groot datalek bij Facebook: check hier of ook uw telefoonnummer erbij is*. URL: <https://www.demorgen.be/tech-wetenschap/groot-datalek-bij-facebook-check-hier-of-ook-uw-telefoonnummer-erbij-is~b24cdba7/> (visited on 08/12/2021).
- [54] Het Laatste Nieuws. *Op deze site kan je zien of ook jouw telefoonnummer te grabbel werd gegooid voor oplichters na groot datalek bij Facebook*. URL: <https://www.hln.be/ihln/op-deze-site-kan-je-zien-of-ook-jouw-telefoonnummer-te-grabbel-werd-gegooid-voor-oplichters-na-groot-datalek-bij-facebook~a24cdba7/> (visited on 08/12/2021).
- [55] Christoforos Ntantogian, Stefanos Malliaros, and Christos Xenakis. “Evaluation of password hashing schemes in open source web platforms”. In: *Computers & Security* 84 (2019), pp. 206–224. ISSN: 0167-4048. DOI: <https://doi.org/10.1016/j.cose.2019.03.011>. URL: <https://www.sciencedirect.com/science/article/pii/S0167404818308332>.
- [56] VRT NWS. *Fuite de données Facebook: l’Autorité de protection des données appelle à la vigilance*. URL: <https://www.vrt.be/vrtnws/fr/2021/04/07/fuite-de-donnees-facebook-l-autorite-de-protection-des-donnees/> (visited on 08/12/2021).
- [57] *OECD Glossary of Statistical Terms*. OECD Publishing, 2008. ISBN: 9789264025561. URL: <https://books.google.be/books?id=wLz-wAEACAAJ>.

- [58] Jay Peters. *Personal data of 533 million Facebook users leaks online*. URL: <https://www.theverge.com/2021/4/4/22366822/facebook-personal-data-533-million-leaks-online-email-phone-numbers> (visited on 08/05/2021).
- [59] Arnold Pindar. *The General Data Protection Regulation*. URL: [https://isotc.iso.org/livelink/livelink/fetch/-8925011/8925029/8925038/17692204/19549786/06\\_-\\_European\\_General\\_Data\\_Protection\\_Law%2C\\_presentation\\_by\\_Arnold\\_Pindar\\_%28ANEC%29.pdf?nodeid=19718099&vernum=-2](https://isotc.iso.org/livelink/livelink/fetch/-8925011/8925029/8925038/17692204/19549786/06_-_European_General_Data_Protection_Law%2C_presentation_by_Arnold_Pindar_%28ANEC%29.pdf?nodeid=19718099&vernum=-2) (visited on 11/16/2020).
- [60] Open Web Application Security Project. *OAT-011 Scraping*. URL: [https://owasp.org/www-project-automated-threats-to-web-applications/assets/oats/EN/OAT-011\\_Scraping](https://owasp.org/www-project-automated-threats-to-web-applications/assets/oats/EN/OAT-011_Scraping) (visited on 08/06/2021).
- [61] ScaleGrid. *2019 Database Trends*. 2019. URL: <https://scalegrid.io/blog/2019-database-trends-sql-vs-nosql-top-databases-single-vs-multiple-database-use/> (visited on 11/01/2020).
- [62] Paul M Schwartz and Daniel J Solove. "The PII problem: Privacy and a new concept of personally identifiable information". In: *NYUL rev.* 86 (2011), p. 1814.
- [63] Florian Smeyers. *Facebook Checker*. URL: <https://facebookleak.netlify.app/> (visited on 08/05/2021).
- [64] Yaroslav Stefinko, Andrian Piskozub, and Roman Banakh. "Manual and automated penetration testing. Benefits and drawbacks. Modern tendency". In: *2016 13th International Conference on Modern Problems of Radio Engineering, Telecommunications and Computer Science (TCSET)*. 2016, pp. 488–491. DOI: [10.1109/TCSET.2016.7452095](https://doi.org/10.1109/TCSET.2016.7452095).
- [65] LATANYA SWEENEY. "k-ANONYMITY: A MODEL FOR PROTECTING PRIVACY". In: *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems* 10.05 (2002), pp. 557–570. DOI: [10.1142/S0218488502001648](https://doi.org/10.1142/S0218488502001648). eprint: <https://doi.org/10.1142/S0218488502001648>. URL: <https://doi.org/10.1142/S0218488502001648>.
- [66] Avast Security News Team. *Google pulls 106 malicious Chrome extensions*. URL: <https://blog.avast.com/google-removes-106-malicious-chrome-extensions-avast> (visited on 08/07/2021).
- [67] Hi Tech. *Massive new LinkedIn data breach hits 92 percent of users; how hacker did it and yes, you should worry*. URL: <https://tech.hindustantimes.com/tech/news/massive-new-linkedin-data-breach-hits-92-percent-of-users-how-hacker-did-it-and-yes-you-should-worry-71625026008296.html> (visited on 07/02/2021).
- [68] Michel van der Ven. *Facebook-lek treft drie miljoen Belgen: liggen jouw gegevens ook op de straatstenen?* URL: [https://datanews.knack.be/ict/nieuws/facebook-lek-treft-drie-miljoen-belgen-liggen-jouw-gegevens-ook-op-de-sstraatstenen/article-news-1719863.html?cookie\\_check=1628755949](https://datanews.knack.be/ict/nieuws/facebook-lek-treft-drie-miljoen-belgen-liggen-jouw-gegevens-ook-op-de-sstraatstenen/article-news-1719863.html?cookie_check=1628755949) (visited on 08/12/2021).
- [69] Wikipedia. *General Data Protection Regulation*. 2020. URL: [https://en.wikipedia.org/wiki/General\\_Data\\_Protection\\_Regulation#Rights\\_of\\_the\\_data\\_subject](https://en.wikipedia.org/wiki/General_Data_Protection_Regulation#Rights_of_the_data_subject) (visited on 11/17/2020).
- [70] Built With. *CMS Usage Distribution on the Entire Internet*. URL: <https://trends.builtwith.com/cms/traffic/Entire-Internet> (visited on 08/13/2021).



- 
- [71] Wix.com. *Preparing Your Wix Site for GDPR*. URL: <https://support.wix.com/en/article/preparing-your-wix-site-for-the-gdpr> (visited on 08/13/2021).
- [72] Wix.com. *Privacy: About Your Customer's Data*. URL: <https://support.wix.com/en/article/gdpr-about-your-customers-data> (visited on 08/13/2021).
- [73] Support — WordPress.com. *Your WordPress.com Site and the GDPR*. URL: <https://wordpress.com/support/your-site-and-the-gdpr/> (visited on 08/13/2021).
- [74] Worldometer. *Belgium Population*. URL: <https://www.worldometers.info/world-population/belgium-population/> (visited on 08/12/2021).
- [75] Worldometer. *Netherlands Population*. URL: <https://www.worldometers.info/world-population/netherlands-population/> (visited on 08/12/2021).
- [76] Mingyi Zhao et al. *Crowdsourced Security Vulnerability Discovery: Modeling and Organizing Bug-Bounty Programs*. URL: <https://aronlaszka.com/papers/zhao2016crowdsourced.pdf> (visited on 08/12/2021).