Use-After-FreeMail: Generalizing the Use-After-Free Problem and Applying it to Email Services

Daniel Gruss[1], Michael Schwarz[1], Matthias Wübbeling[2,3], Simon Guggi[1], Timo Malderle[3], Stefan More[1], Moritz Lipp[1]
[1] Graz University of Technology, Austria
[2] Fraunhofer FKIE, Germany
[3] University of Bonn, Germany

ABSTRACT

Use-after-free is a type of vulnerability commonly present in software written in memory-unsafe languages like C or C++, where a program frees a memory buffer too early. By placing counterfeit structures at the freed memory location, an attacker can leak information or gain execution control upon subsequent access.

In this paper, we show that the concept of use-after-free can be generalized to any environment and situation where resources can be silently exchanged. As an instance of our generalization we demonstrate Use-After-FreeMail attacks. Use-After-FreeMail attacks gather email addresses from publicly available database leaks. The fully automated quantitative analysis brought to light that 33.5% of all free-mail addresses we tested are not valid anymore. In two user studies with 100 and 31 participants we found that 11–19% of users are affected by our attack. In qualitative case studies we investigated what information can be gained in Use-After-FreeMail attacks, e.g., payment information, and how far currently used accounts can be compromised (identity theft). Finally, drawing the connection between mitigations against traditional use-after-free scenarios and the Use-After-FreeMail scenario, we provide a concise list of recommendations to free-mail providers and users as a protection against use-after-free attacks.

CCS CONCEPTS

• **Security and privacy**;

ACM Reference Format:
Daniel Gruss[1], Michael Schwarz[1], Matthias Wübbeling[2,3], Simon Guggi[1], Timo Malderle[3], Stefan More[1], Moritz Lipp[1]. 2018. Use-After-FreeMail: Generalizing the Use-After-Free Problem and Applying it to Email Services. In *ASIA CCS '18: 2018 ACM Asia Conference on Computer and Communications Security, June 4–8, 2018, Incheon, Republic of Korea.* ACM, New York, NY, USA, 15 pages. https://doi.org/10.1145/3196494.3196514

1 INTRODUCTION

In memory-unsafe languages, use-after-free is a common bug, occurring when freeing a memory block too early [4], *i.e.*, the program can still access the memory block through a dangling pointer. An attacker can exploit the vulnerability by reclaiming the same memory block and placing a counterfeit buffer in it [17]. Countermeasures have been proposed that introduce changes on system-level [15], on emulator-level [10], or on compiler-level [37, 60, 64, 72].

Although use-after-free is usually only mentioned in the context of memory-unsafe languages, the concept is far more generic and applies to a wider range of scenarios. Liu et al. [39] already showed that a similar problem exists in sub-domain DNS entries. Swinnen [62] and Bryant [9] exploited a similar attack vector in practice to take over websites. Similarly, Twitter handle reuse [43] and phone number reuse [44] are practical attack vectors. These recent works already indicate that the concept of use-after-free is not restricted to memory-unsafe languages and use-after-free mitigations might likewise be applicable to other scenarios as well.

Over the past decades, email has taken a central role in personal and business activities [26]. We use email not only for communication but *access to an email account* is also used as a means of authentication to modern web services (e.g., password recovery). Many websites do not even allow registration without an email address [7]. Consequently, by gaining access to an email account, an attacker can not only learn about the victim directly [14], but also gain access to other websites the victim uses for personal or professional purposes. While still not widely adopted, two-factor authentication is a proposed solution against these attacks [55].

In recent history, a series of large-scale database leaks occurred, including 450 000 Yahoo email accounts, and 6.5 million LinkedIn accounts. These leaks contain information interesting to an attacker beyond passwords and password hashes, such as user names and email addresses. In this paper, we show that publishing just the email address allows obtaining sensitive information.

In this paper, we present a generalization of use-after-free. We show that use-after-free vulnerabilities exist in any environment and situation where resources can be silently exchanged. We show that use-after-free attacks always work in three phases: A collection phase, an impersonation phase, and an exploitation phase.

We demonstrate Use-After-FreeMail attacks as an instance of our generalized use-after-free vulnerabilities. The vast majority of users own one or more free-mail accounts for their personal, business, spam, and backup emails [30, 33]. Many free-mail providers release old email addresses when they are not used within an expiration period. We show that Use-After-FreeMail attacks always work in the three phases of use-after-free attacks as well.

In the collection phase, the attacker gathers email addresses with dangling references from public database leaks. The attacker automatically tests which email addresses are currently not registered.

ASIA CCS '18, June 4–8, 2018, Incheon, Republic of Korea

ACM ISBN 978-1-4503-5576-6/18/06…$15.00
https://doi.org/10.1145/3196494.3196514

In a fully automated quantitative analysis, we found that 33.5% of the addresses found in online leaks were not valid anymore. In the impersonation phase, the attacker registers a previously expired or deleted email address. In the exploitation phase, the attacker uses the email address to authenticate with other services and to receive emails from other services, e.g., services where the user forgot to update the associated email address.

In a qualitative analysis, we investigated what information can be gained and how far currently used accounts can be compromised in Use-After-FreeMail attacks. In some cases, we were able to recover personal and sensitive information such as payment information and even gain access to still actively used accounts.

We show that Use-After-FreeMail attacks can also be extended to expired domain names, facilitating the same attack vector. In a fully automated attack based on public database leaks, an attacker gathers expired domain names. The attacker registers these domain names and uses them to authenticate with other services via email. We also illustrate how an attacker can directly perform a use-after-free attack based on the domain names especially of former businesses to scam former customers and employees.

Our results indicate that the generalized concept of use-after-free is very powerful and not restricted to memory-unsafe languages. Use-After-FreeMail demonstrates that database leaks are a significant security and privacy issue, even when all information but the email address is removed. We provide a list of recommendations to users and mail providers to prevent Use-After-FreeMail attacks. We conclude that other environments and situations where resources can be silently exchanged should also be investigated to eliminate use-after-free vulnerabilities in these environments.

Contributions. The contributions of this work are:

(1) We show that the concept of use-after-free can be generalized to any environment and situation where resources can be released.
(2) We demonstrate use-after-free attacks on expired free-mail addresses and domain names, obtained through database leaks.
(3) Our quantitative and qualitative analysis show that 33.5% of the leaked email addresses are expired and that an attacker can obtain highly sensitive information such as payment information, and access to currently used accounts of the user.
(4) We provide a concise list of recommendations for free-mail providers and users, to defend against use-after-free attacks.

The paper is organized as follows. Section 2 provides background information. Section 3 introduces a generalization of use-after-free. Section 4 describes our Use-After-FreeMail attack. Section 5 describes how Use-After-FreeMail can be extended to expired domain names. Section 6 provides a quantitative analysis of the problem. Section 7 provides a qualitative analysis based on case studies with volunteers. Section 8 discusses countermeasures, recommendations, and ethical considerations. We conclude our work in Section 9.

2 BACKGROUND

In this section, we explain dangling pointers, use-after-free, database leaks, and the role of emails in the modern web.

2.1 Dangling Pointers

When releasing a resource in memory-unsafe languages, it is essential to invalidate (zero-fill) any existing pointer to this resource. Otherwise, these pointers become dangling pointers. Dangling pointers are the root of use-after-free vulnerabilities [4, 17].

Dangling pointers also exist in other contexts that provide explicit resource allocation and deallocation methods. Liu et al. [39] described DNS entries of sub-domains as pointers to specific virtual machines in the cloud, referenced by their IP address. A dangling DNS entry refers to and exposes the former IP address, which can be allocated by the attacker by renting virtual machines in the same public cloud.

Swinnen [62] showed that dangling DNS entry attacks can be exploited in practice, by taking over a sub-domain of uber.com abusing it to take over user accounts. Kalafut et al. [34] investigated the problem of orphan DNS servers and how they are exploited in practice. Bryant [9] registered the orphan TLD DNS server for `.io`-domains, allowing take over of any `.io` domain.

Several works have observed domain reuse by spammers [13, 28, 29]. Moore and Clayton [48] and later Lever et al. [38] analyzed the problem on a larger scale. They coin the term *residual trust* to describe the credibility that is tied to the expired domain name. Attackers reuse these domain names to exploit this residual trust. Lever et al. [38] also discussed how the residual trust in the domain is related to the residual trust in previously registered mail accounts on that domain. In concurrent work, Borgolte et al. [8] showed that these domain-reuse attacks are significantly more dangerous with modern trust mechanisms, *i.e.*, easy-to-obtain SSL certificates. Independent of our work, they also discovered the connection of these attacks to traditional use-after-free vulnerabilities and named the group of vulnerabilities they found "IP address use-after-free".

Mariconti et al. [43] observed handle reuse on Twitter. When changing the Twitter handle, another user can reuse the released handle. Attackers exploited this for reputation hijacking and impersonation attacks. Martindale [44] showed that Facebook accounts can be taken over through reused phone numbers.

Another source of dangling pointers are human faults (typosquatting) [21] or hardware faults (bitsquatting) [16], modifying the reference from a valid one, to an invalid one. Typosquatting gained much attention by the scientific communities since the first reports about it in 1998 [5, 18, 20, 21, 35, 49, 51, 52, 61]. Both typosquatting and bitsquatting are no reliable dangling pointers, but instead they rely on the introduction of errors to the original and correct reference.

2.2 Mitigating Use-After-Free

Dangling pointers have been studied by programming language designers for 50 years. The idea was to develop memory-safe languages, which technically prevent dangling pointers, mitigating use-after-free bugs and attacks [6, 12, 19, 67]. However, despite these early work's, use-after-free has evolved to a practical threat, being exploited in the wild. Over the past 10 years the number of CVEs reporting use-after-free vulnerabilities increased by two orders of magnitude and is now at approximately 300 per year, whereas the total number of CVEs reported stayed at a constant level [63] (cf. Appendix A). Xu et al. [68] showed that use-after-free

can be reliably exploited in the Linux kernel. This underlines that in practice, use-after-free is far from being solved.

Several works aim to avoid or reduce reusing of virtual memory locations, *i.e.*, once a buffer is deleted, its corresponding pointer becomes invalid and stays invalid (ideally) for the entire program life time [15]. Caballero et al. [10] presented Undangle, a system to detect the creation of dangling pointers. Serebryany et al. [60] presented AddressSanitizer, a system that detects various memory errors, including use-after-free bugs. Lee et al. [37] and Younan [72] presented systems that keep track of all pointers and nullify them when the object is deleted. Kouwe et al. [64] extended their work, building more complete and efficient dangling pointer checks.

All these solutions have in common that they are specific to memory-unsafe programming languages, typically C or C++. As also stated by Kouwe et al. [64], tracking type-unsafe pointer copies is more difficult. As we show in this paper, many systems that are susceptible to use-after-free, do not have the notion of types, type safety, or type-safe pointer copies.

2.3 Database Leaks

A database leak or password leak is a dump of either parts of or an entire database, which is acquired by an attacker and published (or sold) on the Internet. Leaks typically contain user names, email addresses, and passwords. Passwords are often stored as (unsalted) MD5 or SHA1 hashes, or worse as plaintext [36]. With this information, an attacker can probe the source website or other websites to check whether a login with leaked credentials is possible. A successful attacker can impersonate a victim, steal bank account information or other sensitive information and use it to demand ransom from the victim.

The number of leaks increased significantly in recent history (cf. Appendix A). The total number of accounts leaked is currently at 3.9 billion and it grows quickly [31]. A steep increase around the end of 2016 and beginning of 2017 occurred due to several large leaks. Most significantly, as it is the largest leak, is the leak Exploit.In (https://exploit.in) with 593 million accounts, which is a combination list containing user credentials from different web services.

Often, users want to know whether their credentials leaked. Websites like https://haveibeenpwned.com provide a service to check millions of leaked credentials at once, reduced to only the email address. Consequently, users can test whether they are affected, while sensitive information is not exposed.

2.4 How Emails Are Used Today

Since its introduction as a means of communication between two parties, email has evolved to a central part of both professional and personal activities [26]. Today, every user has 1.7 email accounts [56] on average, and this number keeps growing. With the rise of the modern web and its possibilities (e.g., social media, shopping), the need for a web service to contact a user, and the need to verify (to some extent) that a user is real, increased. Today, many websites do not allow registration without an email address anymore [7]. Although today more websites are introducing two-factor authentication, most websites still allow password recovery by simply providing the email address used for registration [7]. Hence, access to an email account does not only provide access to the communication [14], but it is also a way of authentication to third-party websites. Parwani et al. [54] exploited this attack vector through the interaction between Windows Live Messenger and Facebook to takeover arbitrary user accounts without their permission, raising ethical concerns. Mills and Kucherawy [47] observed that this can be a problem and proposed an SMTP extension as a solution.

Most internet users do not have their own mail server. Instead, they use email addresses registered with a free-mail provider [30, 33], allowing email usage free of charge. Users may have multiple accounts with one or more free-mail providers, using them for different purposes and contexts [33]. Still, a non-negligible share of email addresses is hosted on personal or business mail servers. Analyzing the data from the last.fm leak, we find that 0.06% of the hosts appear at least 1000 times in the leak. These 0.06% of providers amount for 93.2% of all registered email addresses, with every provider having 56181.7 addresses in the leak on average. The remaining 99.94% of the hosts amount for only 6.8% of registered email addresses, with every host having 2.4 addresses in the leak on average. We discuss the security implications of use-after-free for personal and business mail servers in Section 5.

3 GENERALIZATION OF USE-AFTER-FREE

In this section, we introduce a generalization of use-after-free that allows to directly identify systems and environments where the use-after-free problem exists.

The core of our generalization is a minimal abstraction of a system with resources:

(1) *The system has the concept of resources.*
Resources (R) can be any entity, e.g., memory buffers, email accounts, servers, or persons. Resources are unique and distinct, *i.e.*, any copy is a resource by itself and distinguishable from the original resource.

(2) *The system has the concept of resource identifiers.*
Resource identifiers (I_R) enable referring to a resource, e.g., pointers, email addresses, domain names, addresses. For any resource, there can be an arbitrary number of resource identifiers. The link between a resource identifier and a resource can be seen as an edge in a directed graph ($I_R \rightarrow R$).

(3) *The system has the concept of identifier holders.*
Identifier holders store resource identifiers for future access to a resource, e.g., variables, business cards, humans, websites, databases. For any resource identifier, there can be an arbitrary number of identifier holders, having a copy of the identifier. The link between identifier holders and resource identifiers can again be seen as an edge in a directed graph ($H_I \rightarrow I_R$).

The problem of use-after-free is *silently exchanging a resource*, *i.e.*, resource identifiers and identifier holders take no notice of the change (cf. Figure 1). That is, the system allows transformation of a directed graph $G : H_I \rightarrow I_R \rightarrow R$ to a different directed graph $G' : H_I \rightarrow I_R \rightarrow R'$, *i.e.*, the edge $I_R \rightarrow R$ is replaced by the edge $I_R \rightarrow R'$. Preventing use-after-free problems in such a system is possible by cutting at least one of the edges of the graph upon resource exchange, using one of the following two approaches:

(1) *Resources cannot be exchanged.*

Figure 1: Use-after-free in an abstract system: Resource R can silently be replaced by resource R' while I_R and H_I remain unchanged.

The edge $H_I \to I_R$ still exists, but the edge $I_R \to R'$ cannot be created. Hence, I_R either has no outgoing edge or it still linked to R through the edge $I_R \to R$. However, this approach is not practical in most scenarios as resources remain in use forever.

(2) *Exchanging resources notifies identifier holders, i.e., the set of identifier holders will be the empty set.*
When R is released, any sub-graph $H_I \to I_R \to R$ is updated such that all edges $H_I \to I_R$ are removed, or the edges $I_R \to R$ are updated to $I_R \to R'$. These are viable solutions for traditional use-after-free situations [10, 15, 37, 64, 72].

The negation of both mitigations, *i.e.*, both an edge $H_I \to I_R$ and an edge $I_R \to R'$ exist, directly yields that the transition illustrated in Figure 1 is possible. Consequently, the link from $H_I \to I_R \to R'$ exists, enabling use-after-free. Use-after-free is possible in any system that can be abstracted to these properties.

3.1 Generalized Use-After-Free Attacks

In a use-after-free attack, an attacker tries to exchange a resource without notification of the identifier holders. The attack works in three steps or phases:

(1) *Collection Phase: Free a resource*
The attacker collects resource identifiers to attack, either by triggering freeing of the resource, or by simply waiting for the resource being released. The result of this step is a dangling resource identifier (a dangling pointer), *i.e.*, $H_I \to I_R \to R$ where R is not allocated anymore.

(2) *Impersonation Phase: Allocate a resource with the same identifier*
In this step, the attacker allocates a resource with the exact same resource identifier, *i.e.*, $H_I \to I_R \to R'$ where R' is the new resource. This resource mimics the original resource with malicious variations to provide the attacker with information or capabilities of the original resource. Common techniques for this step are spraying (*i.e.*, allocating enough resources until one matches), grooming (*i.e.*, exploiting knowledge of allocation algorithms to allocate a matching resource at exactly the right moment), or if the system allows it, by selectively allocating a resource matching the dangling resource identifier.

(3) *Exploitation Phase: Use the outdated resource identifier*
In this step, an identifier holder uses the outdated resource identifier, which now refers to the new counterfeit resource. Consequently, the attacker obtains capabilities or information intended for the original resource. In classical scenarios this phase involves triggering some operation on the outdated resource identifier, or simply waiting until the outdated resource identifier is used by an identifier holder.

Table 1: Systems susceptible to use-after-free.

Resource (R)	Resource Identifier (I_R)	Identifier Holder (H_I)
Memory buffer	Pointer / Address	Variables
Server	DNS entry / (Sub-)domain	Links, databases, human memory
Email account	Email address	Links, third-party websites, databases, address books, human memory
Twitter account	Twitter handle	Links, third-party websites, databases, human memory
Personal Phone	Phone Number	Personal and business address books, third-party websites, human memory
Mailbox	Address	Personal and business address books, human memory
Employee	Office number	Human memory, business cards

3.2 Illustrative Examples of Generalized Use-After-Free

Table 1 shows examples of systems that are susceptible to use-after-free following from our generalization. Attacks on these systems also follow the three steps of our generalized use-after-free attacks.

While use-after-free attacks on buffers in memory-unsafe languages have been studied for several years [4, 17], only recent work has found that DNS entries [9, 39, 62] are susceptible to a similar effect. The DNS entry is the resource identifier which points to a resource, the IP address of the server. In the collection phase of use-after-free attacks on DNS, the attacker finds DNS entries suitable for the attack. In the impersonation phase, the attacker either uses spraying, *i.e.*, allocating a large number of machines such that one of them has the right IP address, or grooming, *i.e.*, performing a targeted allocation or registration. The attacker sets up the server to provide a malicious service instead of the former benign one. In the exploitation phase, the attacker waits for victims to use the attacker-controlled server.

Use-after-free attacks on Twitter handles [43] also work in these three phases. The Twitter handle is the resource identifier which points to a resource, the Twitter profile. In the collection phase, the attacker gathers previously used Twitter handles. In the impersonation phase, the attacker performs a targeted registration of a free Twitter handle. The attacker sets up the profile for malicious activity, e.g., identity theft. In the exploitation phase, the attacker waits for victims to visit the fake profile.

Reuse of phone numbers [44] introduces the same effects. The phone number is the resource identifier pointing to a person, *i.e.*, the resource. In the collection phase, the attacker gathers previously used phone numbers. In the impersonation phase, the attacker uses spraying, *i.e.*, allocating a large number of phone numbers, or grooming, *i.e.*, choosing a specific phone number known to be previously in-use. In the exploitation phase, the attacker uses the phone number to authenticate with other services, e.g., social networks or SMS TAN mechanisms (online banking).

This shows that our generalized description of use-after-free allows analyzing the problem in a wide range of scenarios. In concurrent independent work, Borgolte et al. [8] observed a similar connection between a vulnerabilities arising from IP address reuse and traditional use-after-free attacks, which they hence called “IP address use-after-free vulnerabilities”.

4 USE-AFTER-FREEMAIL

In this section, we study the case of use-after-free in free-mail systems, as an instance of the generalization presented in the previous section. Free-mail systems comply with the three requirements

we identified. The free-mail accounts are the resources. Free-mail accounts can be deleted, or they may even expire automatically after a certain time of inactivity. Subsequently, an attacker can re-register the free-mail account. Throughout this process, the identifier, *i.e.*, the email address, remains unchanged, but now points to a different free-mail account. Hence, the identifier holders (virtually any location where the email address is stored, including the memory of friends and colleagues) take no notice of the change of the underlying free-mail account.

The goal of Use-After-FreeMail attacks is to acquire sensitive user data and access to user accounts, to abuse the data (e.g., bank account information) for malicious purposes, to impersonate the user, or abuse the accounts for any other malicious activity (e.g., criminal activities, abusing online shop accounts, demanding ransom, etc.). As all use-after-free attacks, Use-After-FreeMail attacks work in three phases. In the first phase, the collection phase (cf. Section 4.2), the attacker gathers free-mail addresses of potential victims. In the second phase, the impersonation phase (cf. Section 4.3), the attacker re-registers free-mail accounts of potential victims found in the collection phase. In the third phase, the exploitation phase (cf. Section 4.4), the attacker uses the re-registered free-mail account to authenticate with and use other websites and services. The other websites and services then leak sensitive information directly to the attacker, or provide a means for identity theft and ransom demands.

4.1 Attacker Model

In our attacker model, we make the assumptions that the attacker
(1) is able to acquire a several database leaks from online sources,
(2) has enough abilities or resources to implement several scripts to automate steps of the attack,
(3) can either (a) acquire a large number of phone numbers, or (b) register accounts with common authentication providers (Google, Facebook, Twitter, etc.), *i.e.*, one account per victim,
(4) can either (a) afford a CAPTCHA-solving service, or (b) use Tor to avoid anti-bot mechanisms,
(5) for some attack variants, has enough abilities or resources to run a mail server.

While working on the attack, we found that these assumptions are easily met in practice. Note that despite this generic set of assumptions any attack will still remain site-specific. The attacker will have to adapt to every website to attack and, as discussed in Section 8, websites may have appropriate countermeasures in place.

4.2 Collection Phase

The goal of the collection phase is gathering a significant amount of email addresses of potential victims, *i.e.*, email addresses that are free to register. The search space of email addresses is too large to be searched exhaustively. Hence, an attacker exploits additional information to minimize the search space. In Use-After-FreeMail attacks, an attacker acquires free-mail addresses from database leaks. These addresses have been registered at some point in time, but they may have been deleted or may have expired in the meantime.

The central challenge of the collection phase is to validate which free-mail addresses are free to register. Free-mail providers introduced several mechanisms to prevent abuse of their free-mail addresses and to inhibit automated free-email registration. Mail servers are typically well protected by means of flood protection, grey-listing and generic error messages, to prevent leaking whether an email address is valid or invalid on a large scale. This makes it difficult to determine whether an email address is valid by sending an email or trying to access the IMAP or SMTP server. Trying to login or to register a free-mail through the free-mail provider's website is typically protected using rate limitations and CAPTCHAs.

We found that this central challenge can be overcome by exploiting modern web APIs provided by the website to offer a responsive user interface using AJAX. This is certainly not the only approach that can be pursued in practice, but we found it to be very fast and reliable. When trying to register a free-mail address, the free-mail service provides instant feedback through AJAX requests, to show the user whether an email address is valid and available for registration. We have found the protections applied to these interfaces are much weaker, in some cases we even found the API to be unrestricted. The attacker can now act like an undecided user, typing in the email aliases until all aliases have been tested. We fully automated procedure and hence, automatically determine whether an address is free to register or not, for an entire database leak.

To evaluate the practicality of our approach, we implemented this attack step in a fully automated manner. Section 6 provides a quantitative analysis of the collection phase, where we find that 33.5% of the free-mail accounts are free to register.

4.3 Impersonation Phase

In the impersonation phase, the attacker prepares all resources such that the original resource is fully impersonated by the attacker's resource, *i.e.*, the former free-mail account is replaced by the attacker's free-mail account. The exploitation of this use-after-free situation happens in the following phase. The goal of the impersonation phase is only to successfully register free-mail addresses (from the collection phase). These addresses are ideally still used by identifier holders that remain unaware that the original account was deleted or expired, and that an attacker registered a free-mail account with the same address.

Automated registration of an email can be tricky, because of countermeasures providers installed. Some providers (e.g., Yahoo) require text message verification for registration. Consequently, an attacker would need a new phone number for every victim free-mail address, which may be impractical for some attackers. However, we found that this requirement can be overcome by using accounts from other providers for the registration. That is, you can register a Yahoo account using your Google account instead of a phone number. Furthermore, the registration process is sometimes protected using CAPTCHAs against robots registering free-mail addresses. Solving CAPTCHAs can also be automated using online CAPTCHA-solving services [50]. However, in our analysis we did not use any CAPTCHA-solving service but instead focused on what an attacker can do without such a service.

As the impersonation phase would already invade a user's privacy, e.g., inevitably reading subjects and senders of emails addressed to the former owner of the free-mail account, we only performed this step for users volunteering in our user case studies. The qualitative analysis of these studies is presented in Section 7.

4.4 Exploitation Phase

The goal of the exploitation phase is to gather sensitive and security-critical information and to gain access to accounts of a potential victim, e.g., access to sensitive data of the victim. As in other use-after-free attacks, an attacker waits for an identifier holder to use the outdated resource identifier (*i.e.*, the free-mail address) or triggers a potential identifier holder to do so.

Emails addressed to the original owner of the free-mail address now reach the attacker, e.g., personal emails sent by someone remembering the email address. This highlights a fundamental problem that Use-After-FreeMail uncovered: There is no automated way to alter people's memory upon the deletion or expiration of a free-mail address, rendering some of the defenses against traditional use-after-free attacks inapplicable (cf. Section 2.2). Besides personal emails, also account information and emails for password reset requests will reach the attacker. Using the password-reset functionality from third-party websites allows an attacker to gain access to these account, e.g., social media accounts, bank accounts, online shops, online discussion boards, video streaming services, and many other. The attacker can again automate these checks for popular websites. When implementing these automated checks, we found them to be easier than the free-mail check we described in the previous sections. While this may not cover all accounts that the user has, the attacker can focus on all websites of interest.

For most websites, the password-reset functionality forms a single-factor authentication through the email address used for registration. As this email address now points to the email account of the attacker, the attacker can authenticate with any other website the victim used.

As this phase violates a user's privacy and can also lead to temporary or permanent loss of access to services, we also performed this step only for users volunteering in our user case studies. The results of the case studies are summarized in Section 7.

5 USE-AFTER-FREEMAIL THROUGH EXPIRED DOMAIN NAMES

Similar to the usage of email addresses registered with free-mail providers, it is possible to extend the described attacks to a larger scope. While the attack vector described in Section 4 aims at the user part of an email address, it is further possible to attack the domain part of it. An attacker that is able to re-register a formerly used domain is also able to re-activate all the email addresses of this domain. Previous work hinted at the possibility of using emails of re-registered domains for spam [29, 34] and authentication [38]. However, they focused on taking over other domains, similar as Schlamp et al. [59], whereas our work focuses on authentication with third-party services.

Even though the term *domain owner* is well established, a domain is not really owned by a person or legal entity. As described by the ICANN [32], the domain owner registers the rights to freely dispose of such, but this right is subject to expiry after 1 to 10 years. When a domain expires, the domain owner may renew the domain registration. If the domain is not registered within a time frame of 45 days, *i.e.*, the *Auto Renew Grace Period*, the *Redemption Grace Period* begins. After another 30 days, the domain will be deleted within the next 5 days. If the domain owner did not intervene up to this point, the domain will be publicly available for (re-)registration.

This scenario complies with the three use-after-free requirements as stated in Section 3.1 for the following reasons: The server the domain references is the resource in question that might be silently exchanged after domain expiration. The domain itself is the resource identifier. As outlined above, re-registration of a domain is available by design, leading to the same resource identifier referencing a different resource, *i.e.*, a different server. Subsequently, the re-registered domain can be used to receive emails destined to addresses with the corresponding domain part.

5.1 Collection Phase

In contrast to free-mail accounts, the domain registration status is comparably easy to determine. Based on email addresses in online database leaks, we extract unique domain names. Registered and active domains, *i.e.*, domains that are not expired or still in the auto renew grace period, are located in the zone files of the Domain Name System (DNS) top level domains (TLD). Each Internet registry maintains TLD zone files for the TLDs it is responsible for. Thus, DNS can be utilized to filter active domains as a first step. Available domains and domains in the redemption grace period do not exist in the TLD zone file and remain in the attacker's candidates list. DNS queries for these domains result in *NXDOMAIN* errors. Hence, after this filtering step, only the domains that are most likely to be available for registration remain in the candidates list.

Besides directly available domains, another set of domains is available for registration by an attacker. In the last decade, a market for expired domains emerged. Special online businesses register recently expired domains in order to sell them back to either the former owner, who might just have missed the expiration date, or other interested parties. Strictly speaking, domains that are registered by such a service are registered, but available for transfer. Unlike directly available domains, an attacker has to pay a fee to the current domain owner to get the domain transferred. As there are many separate domain marketplaces, it is rather difficult to get an automated list of these domains. Therefore, we technically limit our experiments to domains that are directly available.

5.2 Impersonation Phase

As the redemption grace period lasts 30 days, expired domains might not be available for registration even though indicated by DNS responses. In this case, the attacker waits for the redemption grace period to pass. The impersonation phase consists of registering the domain and providing resource information, *i.e.*, an IP address as mail exchange (MX) record. The attacker's system is configured to accept all email delivered to it. Thus, emails directed to any mailbox on this domain are delivered to the attacker's mailbox.

Depending on the former domain owner, it might also be possible to impersonate a whole company and scam users of the former website and customers of the former company. An attacker can craft a counterfeit website (impersonation phase) such that visitors believe it is still the original business. This is especially dangerous for small businesses and startups that may go out of business, which in many cases inevitably leads to domain expiration. However, in this paper, we focus on Use-After-FreeMail attacks.

Table 2: Top 10 providers over 607 mio. leaked addresses and their deletion/expiration policies.

Provider	Addresses	% of 607M	Deletion Period	Expiration Period	Re-reg.
Yahoo	146 507 865	24.1%	40–180 days [69]	1 year [70]	✓
Hotmail/Live/Outlook	124 309 619	20.5%	0 days [46]	1 year [45]	✓
mail.ru	90 313 533	14.9%	3 months [41]	≥ 3 months [42]	✓
Gmail	60 221 972	9.9%	2–3 weeks [24]	–	✗ [25]
Yandex	23 588 886	3.9%	1 month [71]	2 years [71]	✓
AOL	21 228 548	3.5%	90 days [53]	90 days [53]	✓
rambler.ru	16 080 401	2.7%	6 months [58]	6 months [58]	✗ [57]
GMX	9 321 442	1.5%	6 months [22]	6 months [23]	✓
web.de	8 296 692	1.4%	1 year [65]	1 year [66]	✓
Lycos	5 870 302	1.0%	2 days [40]	30 days [40]	✓

5.3 Exploitation Phase

The exploitation phase follows the same procedure as for free-mail accounts. However, here an attacker can also facilitate impersonation techniques well-known from domain squatting attacks. The attacker again tries to gather sensitive and security-critical information, for any user that formerly existed on the impersonated domain. Again, personal emails, but also account information and emails to reset the password, for any user will reach the attacker instead. With password-reset functionality from third-party websites, the attacker can then take over control of these accounts.

As this phase would violate the privacy of many users and could lead to loss of access to services, we only performed a quantitative (cf. Section 6) and non-intrusive qualitative analysis (cf. Section 7.3).

6 QUANTITATIVE ANALYSIS

In this section, we provide a quantitative analysis of the potential of Use-After-FreeMail. We will use this quantitative analysis to provide estimates of how many people are potentially affected by these attacks.

6.1 Analysis of Use-After-FreeMail

Analyzing 607 million unique email addresses from different leaks[1] we found that the top 10 providers had a combined share of 83.3% of the addresses leaked. The top 3 providers already amount for 51.1% of addresses in the analyzed leaks. Table 2 shows the number of leaked addresses we found for the most common providers.

As a first step, we investigated the policies of the top providers, as shown in Table 2. We found that most of the top providers allow either deletion of email addresses, typically with a deletion period, or have an expiration period for unused email addresses. Both the deletion period and the expiration period vary widely between different providers. The deletion period is 2 days in the lowest case (Lycos) and at most 6 months (rambler.ru). The expiration period is 30 days in the lowest case (Lycos) and at most 2 years (Yandex). Naturally, this has an impact on the probability that an email address expired. Furthermore, most of the top providers allow re-registration, enabling Use-After-FreeMail attacks.

An exception here are Gmail and rambler.ru, which both do not allow re-registration of deleted email addresses. While this provides better protection for their users, we find that this still does not fully prevent the problem of dangling references, *i.e.*, dangling references on addresses that were never valid in the first place. In fact, we found a small number of Gmail addresses and a larger number of Rambler addresses that were free to register. We suspect that these email addresses were leaked from services that did not validate email addresses. However, these dangling references are just as exploitable as dangling references from account expiration.

[1]Exploit.in, MySpace, Last.fm, AdultFriendFinder, Yahoo; all obtained from public websites via Google searches

Table 3: Collection Phase: Number of addresses free to register and computation time.

Provider	Free	Time/Check	Expected Free	Standard Error
Yahoo	55.69%	0.27 s	81 590 230	1.28%
Hotmail/Live/Outlook	35.59%	1.16 s	44 241 793	1.23%
mail.ru	8.53%	1.74 s	7 703 744	0.72%
Gmail	10.95%	2.59 s	6 594 306	0.80%
Yandex	2.47%	0.11 s	582 645	0.40%
AOL	44.20%	8.60 s	9 383 018	0.80%
rambler.ru	60.04%	6.85 s	9 654 673	1.26%
GMX	35.32%	0.92 s	3 292 333	1.23%
web.de	19.93%	0.61 s	1 653 531	1.03%
Lycos	77.27%	3.96 s	4 535 982	1.08%

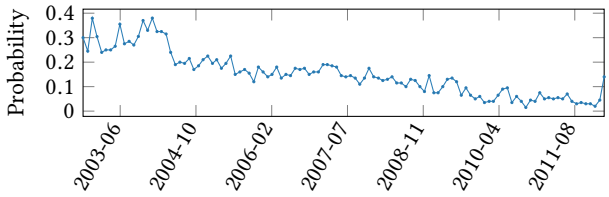

Figure 2: Probability that an email address is free to register as a function of the account registration date found in the last.fm leak.

To investigate how widespread the issue is among different providers, we randomly selected 10 000 email addresses for each of the top 10 providers (cf. Table 2) from public database leaks. We ran the fully automated collection phase on these 100 000 addresses. Our implementation was able to test 7 to 546 addresses per minute, depending on the provider.

Table 3 shows how many free email addresses we found per provider. We computed the number of expected free addresses with the shown standard error values (between 0.40% and 1.29%) and a confidence level of 99%. To assess the risk of email addresses of being exploited, we investigated which email addresses are more attractive for an attacker to target. The most attractive provider for an attacker is Lycos, where more than 77.27% of all addresses found were free to register. We suspect the reason for the high number of free email addresses in the 30-day expiration period. The least attractive provider for an attacker is Yandex, where only 2.47% of the addresses were free to register. However, Yandex was also the fastest service to check for registered addresses, with only 0.11 s per address. The reason for the low number of free email addresses is likely again the expiration period, which is 2 years in this case. The scaled average over all providers is 33%, *i.e.*, an attacker has a 33% chance that a randomly selected email address is free.

Some leaks (e.g., the last.fm leak) provide the account registration date. We investigated the probability that an email address is free to register in relation to the account registration date provided in the leak. Figure 2 shows the probability as a function of the registration date over 22 533 email addresses out of which 3358 (15%) were free

to register. For registration dates before May 2004, on average 30% of the addresses are free to register. We suspect that this value is above average for this data set because email addresses have not been checked automatically as part of the registration process at that time. This lead to addresses in the database that were never valid in the first place. In the time from May 2004 to July 2009, on average 15% of the addresses were free to register. From July 2009 to February 2012 the number of addresses we found free to register gradually decreased from 10% to 2%, with an average of 5%. Note that the correlation between the distribution of hosts in the last.fm leak and our combined data set is 0.86, *i.e.*, the distribution is very similar, and hence, observations on the last.fm data set can be generalized. We conclude that email addresses which were used in the more recent past, are less likely to be free to register.

For registrations in March 2012, we found 14% of the addresses of the addresses in the database to be free to register. We suspect the reason in ongoing registrations at the time of the leak, where preliminary accounts may still contained non-confirmed invalid email addresses. Providing no valid email address constitutes a violation of the terms of use. Usually these accounts are deleted within a short period of time, but this did not happen for the ongoing registrations at the time of the leak yet.

6.2 Use-After-FreeMail based on expired domain names

To evaluate Use-After-FreeMail on domain names of leaked email addresses, it is necessary to pre-filter definitely invalid email addresses. Many services do not check email addresses during the registration process. This leads to invalid email addresses in database leaks. After filtering invalid domain names, the set of potential target domains contains 23 669 300 names in total.

During the evaluation of this huge amount of domain names, we experienced rate limitation on every publicly available DNS resolver we tried. Consequently, we set up our own resolver, which leads to a lower throughput but eliminated the possibility of corrupting results through rate limitation effects.

Resolving all 23 669 300 domains showed that 7 252 062 domains, *i.e.*, 30.6% of the potential target domains, result in the *NXDOMAIN* status. We identified two reasons for this huge portion.

The first reason is that it is possible to have sub-domains in the domain part of an email address. As some TLDs did not allow public registration of second-level domains, registries instead registered the second-level themselves, e.g., co.uk, co.nz, com.br or com.au. The registries then publicly offered third-level domains, *i.e.*, virtually sub-domains, to customers. Therefore, we also checked sub-domains in our evaluation, although the corresponding second-level domain might not have been available for registration. One such example is the domain sverige.hotmail.com. The DNS resolver found that the status of this sub-domain is *NXDOMAIN*, but it is obviously not available for registration. The number of 2nd level domains with NXDOMAIN status is 5 242 711, which is still a huge amount of available domains.

The second reason is that such email addresses were likely provided during a registration process, even though they never existed. Registrations with invalid email addresses inevitably increase the number of available domains found in our evaluation. However, these are still exploitable dangling references, as long as the chosen domain name is valid and not yet registered.

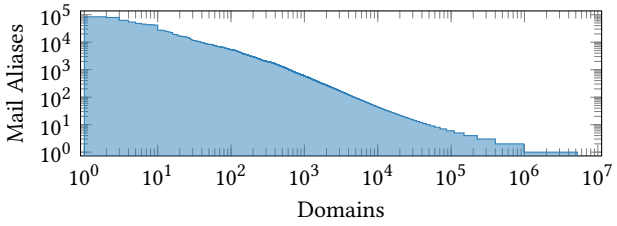

Figure 3: Mail aliases per available domain. Single available domains have up to 83 735 references in database leaks, whereas 4.2 million domains have only a single reference in database leaks.

We filtered the data set which we obtained in our evaluation to only second-level domains, *i.e.*, the most promising domains for an attack. We found that available second-level domains are referenced by 11 575 621 email addresses found in database leaks. Figure 3 shows the number of email addresses per available second-level domain. We can see that most second-level domains are only referenced by a small number of email addresses in database leaks. 4.2 million domains have only a single reference in database leaks. With a growing number of references, the number of domains quickly decreases. Only 38 second-level domains are referenced by more than 10 000 email addresses in database leaks.

The second-level domain with the most references was datingeurope.net,[2] with 83 735 email addresses in database leaks. Hence, by registering this single available second-level domain, an attacker could have gained control over at least 83 735 email addresses that are possibly still referenced by various web services. We provide a detailed case study for this domain in Section 7.3.

7 CASE STUDIES

In this section, we present case studies with users summarizing the types of data that we were able to obtain. First, we performed a quantitative case study to investigate how many users are affected by our attack. We designed the study to have 100 participants to have a maximum standard error below 10 % at a confidence level of 95 %. We found that 60% of the users have an address they don't use anymore and 18% of which, *i.e.*, 11% of all users, are affected by our attack.

Our user case study had a total of 31 participants, which we recruited via University newsgroups and mailing lists, and word-of-mouth. The age range of the participants was from 20 to 53 years. 18 out of 31 participants (which is 58%, very close to the 60% we observed in the quantitative case study described in the last paragraph) provided us with a total of 23 free-mail addresses. We studied the Use-After-FreeMail attack on the 23 free-mail addresses qualitatively. We found 17 of the free-mail addresses were free to register. We performed automated and semi-automated checks against a series of popular websites and found dangling references for 6 free-mail addresses of 6 different users. Hence, for users from

[2]We registered this domain to prevent attackers from using it for malicious purposes (cf. Section 7.3).

the selected University environment, we find that 19% of users are affected (standard error 14%, confidence level 95%). We performed comprehensive case studies with the 6 affected users we found. We summarize the results of these case studies in this section and describe them in detail in Appendix C.

Section 7.3 discusses our case study on datingeurope.net in detail.

7.1 Quantitative Case Study

For our quantitative case study we performed a paid online survey where people answered the question whether they have a free-mail address they do not use anymore, and if so, whether they would provide us with that free-mail address. The participants were from 25 different countries on 5 different continents, 59 from Europe. 60 participants said that they have a free-mail address which they do not use anymore. 36 participants provided us with these free-mail addresses, yielding a total of 108 addresses. By running these addresses against our dataset we found that 14 of these addresses, from 11 different participants, are free to register. We can conclude, with a standard error of 6.2 % and a confidence level of 95 %, that 11% of users are affected by our attack. In Appendix B we provide additional demographic details about the participants.

7.2 User Case Studies

We performed a comprehensive case study with 6 affected users who each had an expired free-mail address from one of 3 different free-mail providers. Out of the 6 email addresses, we found 5 in online leaks, which provided us with a good starting point for the exploitation phase. We attacked these users by re-registering their expired free-mail addresses and using the password reset feature on a series of popular websites. We summarize our findings by categorizing them in 3 categories. A detailed case study for each user can be found in Appendix C.

Personal Information. For all participants, we were able to see a variety of personal details, such as full name, telephone number, and postal address. In addition, we gained access to personal files for half of the participants. We also gained access to social media accounts for half of the participants. As one participant pointed out, an attacker can abuse access to such an account to cause personal or professional damage.

Financial Information. For 2 participants we were able to directly see their bank details. Moreover, one user also had non-public financial information of a nonprofit organisation stored. Besides the direct access to financial information, we had access to the Amazon account of half the participants, allowing us not only to view recent orders but also to place new orders.

Third-party Information. We also gained access to information about third parties from 2 participants. They each had files from a registered association for which they work stored in their private DropBox account. The account for the national automobile club of one participant also revealed personal information about all family members.

7.3 Domain Case Study: datingeurope.net

The domain datingeurope.net was registered from approximately 2008 to 2013. We found this domain to be the domain with the most

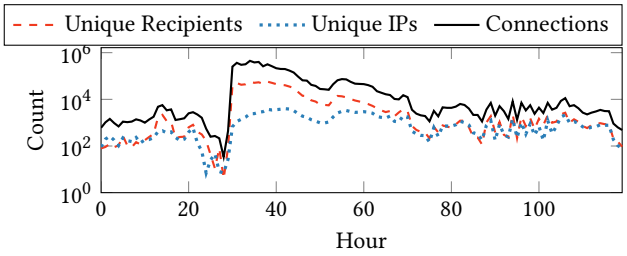

Figure 4: Activity on the mail server over the 5 day time frame of our case study.

dangling email addresses (83 735) in up to 4 database leaks. As our paper reports about the availability of this domain and the dangling references to this domain, we would directly point an attacker to an attractive attack target. Hence, we decided to register this domain, to prevent Use-After-FreeMail attacks through this domain.

We set up an information page recommending to check haveibeenpwned.com as well as all accounts to eliminate any dangling references that enable these attacks. We set up a mail server immediately rejecting any emails to datingeurope.net. This guarantees that we do not receive any emails, subjects or other meta-data beyond the sender email address and the IP address of the connecting mail server. Over a period of 5 days, we logged accesses to the web server (time, IP address) and the mail server (time, IP address of sending mail server, email address, login name).

On the web server we only observed 401 connections, out of which 105 were non-malformed benign requests and 148 non-malformed malicious requests. 111 out of 148 malicious requests tried to access a database dump, expected to be in the web root folder.

Figure 4 shows the activity on the mail server over the 5 day time frame. On the mail server, we received the first connection less than 3 minutes after the domain registration. We observed a total of 5 496 304 connections over the 5 days, *i.e.*, 13 connections per second on average, from 44 011 unique IP addresses, trying to deliver 1 028 719 mails to 85 690 recipients. The top IP address accounted for 24 607 of the requests. While there was only little activity on the server within the first hours, we experienced a significant jump in activity after 30 hours, likely because of delayed DNS updates. The peak activity was 34 hours after registration with 446 287 connections in one hour, *i.e.*, 124 connections per second, from 2398 unique IP addresses.

Out of the 83 736 addresses we found in online leaks, emails were sent to 83 599. The remaining 137 email addresses do not show up in our log files. However, in our log files we found 2091 addresses which do not show up in any online leak we found.

We suspect that most of the emails would have been spam emails. However, our results clearly show the viability of this attack.

8 DISCUSSION

We have seen that use-after-free is a far more generic and omnipresent problem than anticipated. Hence, we need to raise awareness that use-after-free may occur in almost any system. With Twitter and DNS entry use-after-free being exploited in the wild, it

is very likely that free-mail use-after-free is also being exploited in the wild. Consequently, we need to take action to prevent further damage to users, *i.e.*, we need to cut at least one of the links shown in Figure 1, or make resource reuse impossible. With our work, we want to tie different areas of research (*i.e.*, use-after-free on memory buffers and objects, domains, and emails) together. Despite the fact that different forms of use-after-free attacks may have vastly different properties (*i.e.*, it may be easy to find an expired mail address, whereas dangling pointers have fortunately become a more rare occasion in modern software), it will help to retrofit defenses from one area of research to other areas.

In the following, we discuss recommendations to reduce the problem of use-after-free on emails, as well as ethical considerations regarding this work and its responsible disclosure.

8.1 Applying Existing Countermeasures against Traditional Use-After-Free Attacks to Use-After-FreeMail

After observing the similarities between traditional use-after-free situations and our use-after-free situation for mail addresses, the question arises whether countermeasures could be adapted directly to mitigate our attacks. An important observation is that in contrast to the traditional use-after-free situations our real-world situation does not have the notion of a "program life time" after which all virtual memory of the program is released anyway. Instead, an email address (virtually a pointer) can be referenced forever. Dhurjati et al. [15] recommended that once a buffer is deleted, the corresponding pointer becomes and stays invalid. Applying this idea to mail addresses, mail addresses must become and stay invalid forever, once the mailbox it refers to is deleted.

Several works propose to solve the problem using compiler extensions, e.g., Caballero et al. [10] detect the creation of dangling pointers at runtime, Serebryany et al. [60] to detect, besides other bugs, use-after-free bugs at compile time, other works [37, 64, 72] keep track of all pointer copies and nullify when the corresponding memory block is released. While it would be an ideal solution to nullify all copies of an email address when the corresponding mailbox is deleted, this poses practical challenges, since the copies are not stored in a predictable defined way, but can have any form and maybe are stored in human memory. However, as we discuss in the following subsections, several ideas can be adapted to at least partially resolve the Use-After-FreeMail problem. This highlights that mitigating use-after-free in scenarios which were unconnected prior to our work, should not be seen as unrelated work, but instead the scientific community could benefit by connecting research in these areas.

8.2 Recommendations for Providers and Web Services

Mills and Kucherawy [47] already proposed a require-recipient-valid-since header field or SMTP extension in RFC 7293. They suggest that a sender adds a timestamp to the email header stating when the ownership of the mailbox was last known to the sender. If the receiving mail server notices that the mailbox ownership changed since then, the email bounces to the sender. Although this solution is not complete (e.g., use-after-free on domains), it would significantly raise the bar to perform Use-After-FreeMail attacks. While there are implementations for RFC 7293 [1], they are not widely adopted in practice [2, 3]. We also did not observe any active RFC 7293 mitigations on any mail server in analysis.

Non-expiring resource identifiers, as suggested by Dhurjati et al. [15], may be a viable solution in many places. Especially for email addresses, it might be the most complete solution, as it fully eliminates use-after-free. For instance, Google and rambler.ru do not release resource identifiers after users delete their accounts (*i.e.*, free the resource). Instead, the resource identifier remains invalid, making it impossible for attackers to use it for use-after-free attacks. Hence, our most important recommendation is that providers should follow this example and keep once-registered email aliases marked unavailable forever.

In a different direction, services could introduce mechanisms to block Use-After-FreeMail attacks. First, password recovery is significantly more difficult if the attacker also needs to know the user ID or user name. This would also confine the extent of automation that our current implementation has. Second, services could provide the option of two-factor authentication and encourage users to use it. Many of the attacks we performed in the exploitation phase (cf. Section 7), would not have been possible with two-factor authentication. Still, two-factor authentication is also not a complete solution, as for communication purposes the outdated email address might still be used by someone.

However, in all variants, a remaining problem is that free-mail providers may go out of business, selling their domains and thus deleting all email aliases. The problem also remains for private and mail servers where the domain expired. In this case, the attacker can still mount an attack as illustrated in Section 5.

A different approach to secure users against use-after-free is to verify the existence of a user's email address regularly, similar to the regular checks for dangling pointers Caballero et al. [10] proposed. For this purpose, web services could use the same techniques as we did in this paper (cf. Section 4), or alternatively use regular emails sent to the user, e.g., newsletters, invoices. If an email account is expired, emails cannot be delivered successfully, but instead a bounce reply is generated. As an immediate step, web services and providers should take more care when receiving a bounce reply. Bounce replies can be automatically parsed to detect whether an email address has become invalid. Once an email address becomes invalid, it is not the original authentication factor anymore. Hence, password or account recovery through this email address should not be allowed anymore. A similar approach is also pursued by the Certified Senders Alliance [11].

Another approach is to track user activity through unique user-identifying links included in sent emails, e.g., newsletters, invoices. Whenever a user clicks on such a link, the server learns that the user still uses the email account. If there is no user activity through these links over a certain amount of time, an additional verification email could be sent to re-validate the email address.

8.3 Recommendations for Users

Some users of our case study let their free-mail addresses expire or actively deleted them as a reaction to a database leak. They were not aware that this increases their own attack surface. Users should try

to keep unused free-mail addresses, to prevent this class of attacks. Additionally, some users were unaware that their former free-mail address was still used by some web services. We recommend users to review and check all web services they registered with, to update their email address to a new email address. However, this is prone to human error, forgetting a few web services and thus remaining susceptible to use-after-free.

Enabling two-factor authentication can already contribute significantly to account security. However, as discussed in Section 8.2, this requires support by the web service provider.

8.4 Use-After-FreeMail on Non-Public Providers

Naturally, Use-After-FreeMail is not limited to free or public mail providers, but it also exists for non-public providers. During our early investigations, we found one incident where the official Twitter account of a national political party sitting in several councils at that time was referencing an invalid email address on a non-public mail host. This case may have been exploitable by registering as a new party member and choosing this email alias. However, the problem was quickly resolved after we informed the party.

8.5 Ethical Considerations

Working with sensitive user data raises ethical concerns, and while our institution does not have an ethics board, we tried to address ethical concerns throughout our study. First of all, for ethical reasons, we did not perform any of the experiments on email addresses we learned from leaks, but only for persons voluntarily participating in our study. Second, the personal (and potentially sensitive) data gathered in the experiments was only visible for two of the authors, while the other authors only had access to blinded information. All non-anonymized data was deleted when the study was completed. Third, the participants of our study provided their data voluntarily and explicitly gave permission to perform the study. The participants were informed that they had the option to revoke this permission and leave the study at any time. In this case, the personal data gathered would have been deleted immediately. Any security or privacy issues we found, especially concerning accounts we compromised, were immediately discussed with the study participants, helping them to change their passwords, remove the outdated information and protect their accounts more thoroughly. In several cases, we recommended the study participants to enable two-factor authentication as a protection mechanism and to review and check any web service registration to see where the email address should be updated. In all cases, the study participants were thankful for our recommendations and the help in protecting their accounts. At the end of the study, any personal data we gathered was destroyed.

Another consideration is that publishing our insights will allow users to assess the risks of outdated information in their online accounts. Consequently, users can update their information and stay safe from these attacks even when data breaches occur.

Gathering information about individuals accessing dating portals is usually highly sensitive information (e.g., the Ashley Madison leak). For the 5-day case study on datingeurope.net, we took care not to gather any data that would violate the privacy of visitors or former users. We avoided gathering personal and sensitive information by setting up the web server in a way that it only logs accesses to a static page, but does not accept or log any data transmitted by the visitor. Our mail server rejected any attempt of sending or receiving emails, *i.e.*, no email subject, headers, or body were transmitted to our server. The mail server logged only time, IP address and sender and recipient email address, as well as the login name for login attempts. After completion of our study, we deleted any digital non-anonymized data gathered in these 5 days.

8.6 Responsible Disclosure

We initiated responsible disclosure with 5 free-mail providers to eliminate the problem through the fixes we proposed in Section 8.2. We also initiated responsible disclosure with 5 websites where we expect, based on our observations in the user case studies, hardening of password-recovery procedures to help preventing exploitation of use-after-free. All parties quickly responded and acknowledged the general problem. We grouped the answers into different categories:

Helpful. The national automobile club we contacted, responded that they will integrate some of the countermeasures we proposed.

Informative. Yahoo, Amazon, and DropBox promptly responded that they are aware of the problem and are continuously working on improving the protection of their interfaces.

Irrelevant. Mail.ru stated that they only release email addresses after 6 years and hence, consider these attacks irrelevant. Microsoft responded that it is the (former) users responsibility to update any dangling references, not theirs. We received similar comments from other parties.

9 CONCLUSION

In this paper, we showed that use-after-free is a more general and widespread phenomenon than it was known. We introduced a new generalization of use-after-free and showed that this generalization allows to determine whether a system or environment enables use-after-free. We demonstrate our generalization with the specific case of Use-After-FreeMail attacks, which gather free-mail addresses from online database leaks. In a fully-automated quantitative analysis, we found that 33.5% of all free-mail addresses we tested, were not valid anymore. In two user studies with 100 and 31 participants we found that 11–19% of users are affected by our attack. In case studies we were able to recover payment information and to obtain access to actively used accounts of the user. We showed the connection between research on use-after-free mitigation in different areas and provide a concise list of recommendations of immediate and long-term actions to mail providers, web service providers, and users to prevent the exploitation of use-after-free.

ACKNOWLEDGMENTS

We would like to thank our anonymous reviewers for their valuable feedback. We also would like to thank the volunteers of our user study. This work has been supported by the Austrian Research Promotion Agency (FFG), the Styrian Business Promotion Agency (SFG), the Carinthian Economic Promotion Fund (KWF) under grant number 862235 (DeSSnet).

REFERENCES

[1] 2018. Authentication-Results-Python. (2018). https://launchpad.net/authentication-results-python Retrieved on March 06, 2018.

[2] 2018. Popularity contest statistics for authres. (2018). https://qa.debian.org/popcon.php?package=authres Retrieved on March 06, 2018.

[3] 2018. Popularity contest statistics for postfix. (2018). https://qa.debian.org/popcon.php?package=postfix Retrieved on March 06, 2018.

[4] Jonathan Afek and Adi Sharabani. 2007. Dangling pointer: Smashing the pointer for fun and profit. In *Black Hat 2007 Briefings*.

[5] Pieter Agten, Wouter Joosen, Frank Piessens, and Nick Nikiforakis. 2015. Seven months' worth of mistakes: A longitudinal study of typosquatting abuse. In *NDSS'15*.

[6] DM Berry, L Chirica, JB Johnston, DF Martin, and A Sorkin. 1973. On the time required for retention. *ACM Sigplan Notices* 8, 11 (1973), 165–178.

[7] Joseph Bonneau and Sören Preibusch. 2010. The Password Thicket: Technical and Market Failures in Human Authentication on the Web.. In *The 9th Workshop on the Economics of Information Security*.

[8] Kevin Borgolte, Tobias Fiebig, Shuang Hao, Christopher Kruegel, and Giovanni Vigna. 2018. Cloud Strife: Mitigating the Security Risks of Domain-Validated Certificates. In *NDSS'18*.

[9] Matthew Bryant. 2017. The .io Error – Taking Control of All .io Domains With a Targeted Registration. (July 2017). https://thehackerblog.com/the-io-error-taking-control-of-all-io-domains-with-a-targeted-registration/

[10] Juan Caballero, Gustavo Grieco, Mark Marron, and Antonio Nappa. 2012. Undangle: early detection of dangling pointers in use-after-free and double-free vulnerabilities. In *Proceedings of the 2012 International Symposium on Software Testing and Analysis*. ACM, 133–143.

[11] Certified Senders Alliance (CSA). 2016. Admission Criteria - Pertaining to the Participation of Bulk Mailers in the Certified Senders Alliance (CSA). (2016). https://certified-senders.org/wp-content/uploads/2017/07/CSA_Admission_Criteria.pdf

[12] LM Chirica, TA Dreisbach, DF Martin, JG Peetz, and A Sorkin. 1973. Two PARALLEL EULER run time models: The dangling reference, impostor environment, and label problems. In *Proceedings of a symposium on High-level-language computer architecture*. ACM, 141–151.

[13] Na Dai, Brian D Davison, and Xiaoguang Qi. 2009. Looking into the past to better classify web spam. In *Proceedings of the 5th international workshop on adversarial information retrieval on the web*.

[14] Anupam Das, Joseph Bonneau, Matthew Caesar, Nikita Borisov, and XiaoFeng Wang. 2014. The Tangled Web of Password Reuse.. In *NDSS'14*.

[15] Dinakar Dhurjati and Vikram Adve. 2006. Efficiently detecting all dangling pointer uses in production servers. In *Dependable Systems and Networks, 2006. DSN 2006. International Conference on*. IEEE, 269–280.

[16] Artem Dinaburg. 2011. Bitsquatting: DNS hijacking without exploitation. In *Black Hat 2011 Briefings*. https://media.blackhat.com/bh-us-11/Dinaburg/BH_US_11_Dinaburg_Bitsquatting_WP.pdf

[17] Igor Dobrovitski. 2003. Exploit for CVS double free () for linux pserver. (2003).

[18] Benjamin Edelman. 2003. Large-scale registration of domains with typographical errors. (2003). https://cyber.harvard.edu/archived_content/people/edelman/typo-domains/

[19] Charles N. Fischer and Richard J. LeBlanc. 1980. The implementation of runtime diagnostics in Pascal. *IEEE Transactions on Software Engineering* 4 (1980), 313–319.

[20] G Gee and P Kim. 2011. *Doppelganger Domains*. Technical Report. Godai Group.

[21] Dara B Gilwit. 2003. The Latest Cybersquatting Trend: Typosquatters, Their Changing Tactics, and How to Prevent Public Deception and Trademark Infringement. *Washington University Journal of Law & Policy* 11, 1 (2003), 267–294.

[22] GMX. 2017. Deleting an Account. (2017). https://support.gmx.com/account/managing/delete.html

[23] GMX. 2017. General terms and conditions. (2017). https://service.gmx.net/de/cgi/g.fcgi/products/mail/agb

[24] Google. 2017. Recover your Google Account. (2017). https://support.google.com/accounts/answer/6236295

[25] Google. 2017. User Agreement. (2017). https://support.google.com/mail/answer/56256

[26] Catherine Grevet, David Choi, Debra Kumar, and Eric Gilbert. 2014. Overload is overloaded: email in the age of Gmail. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*.

[27] Miniwatts Marketing Group. 2017. World Internet Users Statistics and 2017 World Population Stats. (2017). http://www.internetworldstats.com/stats.htm

[28] Zoltan Gyongyi and Hector Garcia-Molina. 2005. Web spam taxonomy. In *First international workshop on adversarial information retrieval on the web (AIRWeb 2005)*.

[29] Shuang Hao, Matthew Thomas, Vern Paxson, Nick Feamster, Christian Kreibich, Chris Grier, and Scott Hollenbeck. 2013. Understanding the domain registration behavior of spammers. In *Proceedings of the 2013 conference on Internet measurement conference*.

[30] John B Horrigan. 2008. Use of cloud computing applications and services. (Sept. 2008). http://www.pewinternet.org/2008/09/12/use-of-cloud-computing-applications-and-services/

[31] Troy Hunt. 2017. Have I been pwned? Check if your email has been compromised in a data breach. (2017). https://haveibeenpwned.com

[32] ICANN - Internet Corporation for Assigned Names and Numbers. 2017. Life Cycle of a Typical gTLD Domain Name. (2017). https://www.icann.org/resources/pages/gtld-lifecycle-2012-02-25-en

[33] Iulia Ion, Niharika Sachdeva, Ponnurangam Kumaraguru, and Srdjan Čapkun. 2011. Home is safer than the cloud!: privacy concerns for consumer cloud storage. In *Proceedings of the Seventh Symposium on Usable Privacy and Security*.

[34] Andrew J Kalafut, Minaxi Gupta, Christopher A Cole, Lei Chen, and Nathan E Myers. 2010. An empirical study of orphan DNS servers in the internet. In *Proceedings of the 10th ACM SIGCOMM conference on Internet measurement*.

[35] Mohammad Taha Khan, Xiang Huo, Zhou Li, and Chris Kanich. 2015. Every second counts: Quantifying the negative externalities of cybercrime via typosquatting. In *S&P'15*.

[36] Georgios Kontaxis, Elias Athanasopoulos, Georgios Portokalidis, and Angelos D Keromytis. 2013. SAuth: protecting user accounts from password database leaks. In *CCS'13*.

[37] Byoungyoung Lee, Chengyu Song, Yeongjin Jang, Tielei Wang, Taesoo Kim, Long Lu, and Wenke Lee. 2015. Preventing Use-after-free with Dangling Pointers Nullification.. In *NDSS'15*.

[38] Chaz Lever, Robert Walls, Yacin Nadji, David Dagon, Patrick McDaniel, and Manos Antonakakis. 2016. Domain-Z: 28 registrations later measuring the exploitation of residual trust in domains. In *S&P'16*.

[39] Daiping Liu, Shuai Hao, and Haining Wang. 2016. All Your DNS Records Point to Us: Understanding the Security Threats of Dangling DNS Records. In *CCS'16*.

[40] Lycos. 2017. Lycos Mail Terms of Service. (2017). http://info.lycos.com/resources/terms-of-service/mail/

[41] Mail.ru. 2017. Deleting your account. (2017). https://help.mail.ru/engmail-help/registration/delete

[42] Mail.ru. 2017. User Agreement. (2017). https://help.mail.ru/engmail-help/UA

[43] Enrico Mariconti, Jeremiah Onaolapo, Syed Sharique Ahmad, Nicolas Nikiforou, Manuel Egele, Nick Nikiforakis, and Gianluca Stringhini. 2017. What's in a Name?: Understanding Profile Name Reuse on Twitter. In *WWW'17*.

[44] James Martindale. 2017. I kinda hacked a few Facebook accounts using a vulnerability they won't fix. (2017). https://medium.com/@jkmartindale/i-kinda-hacked-a-few-facebook-2f5669794f79

[45] Microsoft. 2017. Deleted Hotmail Account due to Inactivity. (2017). https://answers.microsoft.com/en-us/outlook_com/forum/oemail-orestoremail/d/252611c1-94d6-4b48-974c-005038ea46f7

[46] Microsoft. 2017. How to close your Microsoft account. (2017). https://support.microsoft.com/en-us/help/12412/microsoft-account-how-to-close-account

[47] W Mills and M Kucherawy. 2014. *The Require-Recipient-Valid-Since Header Field and SMTP Service Extension*. RFC 7293. Internet Requests for Comments.

[48] Tyler Moore and Richard Clayton. 2014. The ghosts of banking past: Empirical analysis of closed bank websites. In *FC'14*.

[49] Tyler Moore and Benjamin Edelman. 2010. Measuring the perpetrators and funders of typosquatting. In *FC'10*.

[50] Marti Motoyama, Kirill Levchenko, Chris Kanich, Damon McCoy, Geoffrey M Voelker, and Stefan Savage. 2010. Re: CAPTCHAs-Understanding CAPTCHA-Solving Services in an Economic Context.. In *USENIX Security Symposium*.

[51] Nick Nikiforakis, Marco Balduzzi, Lieven Desmet, Frank Piessens, and Wouter Joosen. 2014. Soundsquatting: Uncovering the use of homophones in domain squatting. In *International Conference on Information Security*.

[52] Nick Nikiforakis, Steven Van Acker, Wannes Meert, Lieven Desmet, Frank Piessens, and Wouter Joosen. 2013. Bitsquatting: Exploiting bit-flips for fun, or profit?. In *WWW'13*.

[53] Oath Inc. 2017. Terms of Service. (2017). http://legal.aol.com/terms-of-service_full-terms/

[54] Tarun Parwani, Ramin Kholoussi, and Panagiotis Karras. 2013. How to Hack into Facebook without being a Hacker. In *WWW'13*.

[55] Thanasis Petsas, Giorgos Tsirantonakis, Elias Athanasopoulos, and Sotiris Ioannidis. 2015. Two-factor authentication: is the world ready?: quantifying 2FA adoption. In *EuroSec'15*.

[56] Sara Radicati and Quoc Hoang. 2015. *Email statistics report, 2011-2015 – Executive Summary*. Technical Report. The Radicati Group, Inc. http://www.radicati.com/wp/wp-content/uploads/2015/02/Email-Statistics-Report-2015-2019-Executive-Summary.pdf

[57] Rambler. 2017. How to delete an account. (2017). https://help.rambler.ru/id/id-manage/1200/

[58] Rambler. 2017. Rules for the use of projects and services. (2017). https://help.rambler.ru/legal/1430

[59] Johann Schlamp, Josef Gustafsson, Matthias Wählisch, Thomas C Schmidt, and Georg Carle. 2015. The abandoned side of the Internet: Hijacking Internet

resources when domain names expire. In *International Workshop on Traffic Monitoring and Analysis*.
[60] Konstantin Serebryany, Derek Bruening, Alexander Potapenko, and Dmitriy Vyukov. 2012. AddressSanitizer: A Fast Address Sanity Checker.. In *USENIX Annual Technical Conference*. 309–318.
[61] Jeffrey Spaulding, Shambhu Upadhyaya, and Aziz Mohaisen. 2016. The landscape of domain name typosquatting: Techniques and countermeasures. In *11th International Conference on Availability, Reliability and Security (ARES)*.
[62] Arne Swinnen. 2017. SGX Secure Enclaves in Practice: Security and Crypto Review. (June 2017). https://www.arneswinnen.net/2017/06/authentication-bypass-on-ubers-sso-via-subdomain-takeover/
[63] The MITRE Corporation. 2017. Common Vulnerabilities and Exposures. (2017). http://cve.mitre.org/
[64] Erik van der Kouwe, Vinod Nigade, and Cristiano Giuffrida. 2017. DangSan: Scalable Use-after-free Detection.. In *EuroSys'17*.
[65] Web.de. 2017. Delete Account. (2017). https://hilfe.web.de/premium/vertraege/loeschen.html
[66] Web.de. 2017. General terms and conditions. (2017). https://agb.web.de/registrierungspflichtig/AGB/20170622
[67] Nikolaus Wirth. 1976. Pascal Newsletter. (Sept. 1976), 29 pages.
[68] Wen Xu, Juanru Li, Junliang Shu, Wenbo Yang, Tianyi Xie, Yuanyuan Zhang, and Dawu Gu. 2015. From collision to exploitation: Unleashing use-after-free vulnerabilities in linux kernel. In *CCS'15*.
[69] Yahoo Holdings Inc. 2017. Close your Yahoo Account. (2017). https://help.yahoo.com/kb/SLN2044.html
[70] Yahoo Holdings Inc. 2017. Reasons Yahoo deactivates or deletes an account. (2017). https://help.yahoo.com/kb/SLN3057.html
[71] Yandex. 2017. User Agreement for Yandex Services. (2017). https://yandex.com/legal/rules/
[72] Yves Younan. 2015. FreeSentry: protecting against use-after-free vulnerabilities due to dangling pointers.. In *NDSS'15*.

A CVE AND DATABASE LEAK STATISTICS

Figure 5 shows how the total number of CVEs and and the number of use-after-free CVEs evolved over the past 10 years. We can see that the number of use-after-free CVEs increased, whereas the total number of CVEs remained at approximately the same level. Figure 6 shows the number of accounts which have been leaked in online database leaks over the past 10 years. We can see a steep increase at the end of 2016, due to multiple massive leaks.

B ONLINE SURVEY DEMOGRAPHICS

In this section, we provide demographic statistics on the participants of our online survey user study. Figure 7 compares the distribution of the participants over continents with the distribution of internet users over continents. We can see that in our data set Europe is overrepresented, whereas the other regions are slightly underrepresented. Figure 8 shows the age distribution of the participants. We can see that the group from age 26 to 35 has the most participants, whereas the group with age 56 and above has the least participants.

C USER CASE STUDIES

In this section, we provide the 6 user case studies that we summarized in Section 7.2.

C.1 User Case Study #1

The participant had a Yahoo account that expired at an unknown date due to inactivity. The email address occurs in two database leaks, the last.fm leak and the Dropbox leak, both from 2012 and publicly released in 2016. While performing our study, we also received emails with job offers revealing specific former job interests of the participants.

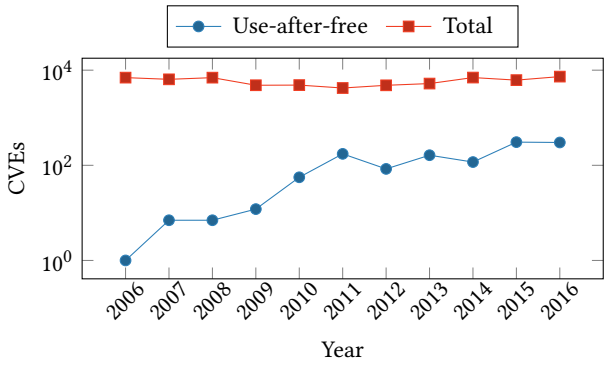

Figure 5: Total CVEs and use-after-free CVEs per year. While the number of total CVEs slightly varied around the same level in the past 10 years, the number of use-after-free CVEs increased significantly.

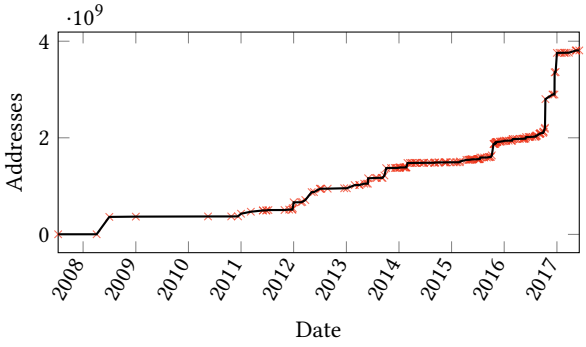

Figure 6: Accumulated number of accounts in online leaks over time.

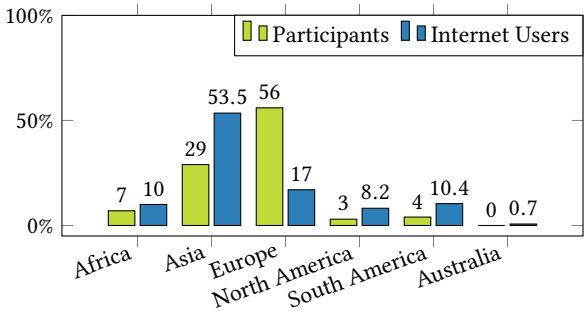

Figure 7: Distribution of participants and internet users over continents [27].

By using the password recovery functions on a series of websites we were able to compromise 6 online accounts of the participant. Together with the study participant, we studied thoroughly what an attacker could have done. In all cases, the account and password recovery only took seconds and was in most cases not protected by any CAPTCHA or anti-bot mechanism.

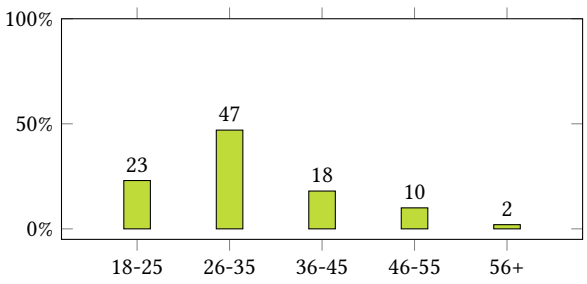

Figure 8: Age distribution of the participants (grouped by years).

The participant protected the current Google account with two-factor authentication. In this case, Google's password-reset procedure requires the knowledge of two factors for password reset. That is, for instance, answering security questions and using the recovery email address. If an attacker guesses the answer to a security question correctly, the attacker would have to file a written explanation why access to the account was lost. Only after this step and an additional delay of a few hours or days, Google may provide the attacker with a password-reset link. Although the Yahoo address was the recovery email address, we believe that the security questions may prevent an attacker from compromising the account.

We were able to recover access to the Amazon account by correctly guessing the postal code, which could be found in other online accounts. Within the Amazon account, we were able to see previous orders and further personal details, such as the full postal address and telephone number.

In the Flipboard account of the participant, we were able to see reading preferences. Similarly, the last.fm account showed the exact music tracks the participant listened to in the past. The study participant remarked that the compromised account could be used to cause personal or professional damage. An attacker could spoof the participant's usage of last.fm, such as listening to music expressing dubious political views.

The Apple account (AppleID) was deactivated by the participant several years ago. Still, we were able to re-enable it and read personal details that have not been deleted from the account. By knowing the AppleID, we also gained access to iCloud and iTunes.

Finally, in the Dropbox account of the participant, we were able to see personal information and information on the last used devices. More significantly, we found financial information stored in files, including financial information of a nonprofit association.

While going through the accounts with us, the study participant already updated the email addresses. We also recommended reviewing all other accounts to find references to the outdated email address and update it. The study participant now plans to keep the email address we re-registered for some time to prevent an attacker from performing this attack.

C.2 User Case Study #2

The participant had a Yahoo account until autumn 2016 and deleted the Yahoo account deliberately, in turn of media reports on Yahoo confirming a leak of 500 million user accounts. The participant was not aware that this might increase his exposure to attackers. Furthermore, the participant did not update the email address in several online accounts. Yahoo has a 40 days deletion policy [69], hence, by the end of 2016 the email address became available again. The email address occurs in several database leaks, including Dropbox, LinkedIn, Stratfor, and tumblr.

We performed checks against a series of popular websites. For many services, the participant did not use two-factor authentication, either because it was opt-in or the website did not support it at all, allowing us to hijack the corresponding accounts. These accounts include the Amazon account, the Dropbox account, the eBay account, but also the account at the national automobile club. Together with the user we analyzed what an attacker can gain from these accounts in more detail.

By resetting the DropBox password we were able to see personal information like full name and phone number, but also all personal files stored in the DropBox folders. The DropBox account was still actively used by the study participant to store and archive data of a registered association. The study participant found this data to be uncritical and loss of access to the DropBox would merely be a nuisance.

LinkedIn either deleted the account or the participant had changed the email address. Hence, no attack on the LinkedIn account was possible.

Even before recovering the password, eBay already provided us with 5 digits of the telephone number. The full telephone number, bank details, and personal information were visible after login. In eBay it is possible to see all previous orders, conveying possibly very personal or sensitive information. The study participant said would be extremely unpleasant if an attacker would see these previous orders and the attached information. The study participant also found an attacker being capable of placing new orders to be unsettling.

Similarly, information could be gained from the Amazon account. After login, Amazon shows the full telephone number, person details, as well as partial bank details. Similar as in the case of eBay, Amazon provides an attacker with full information on all previous orders, including payment information but also conveying possibly very personal or sensitive information. Amazon hinders the placement of new orders to new addresses, but we believe an attacker can find a work-around to still place orders. The study participant again remarked that such attacks would be very unpleasant and unsettling.

In the account of the national automobile club we were able to obtain date of birth, telephone number, full name and address of the participant, all family members and their corresponding age, all vehicles registered including information on license plates, and finally, full bank account details, and credit card information.

While performing this analysis, the study participant already updated the email address in the tested accounts. Furthermore, we provided the study participant with the recommendation to review all other accounts to find references to the outdated email address and update it. The study participant finally remarked to plan to re-register the old Yahoo email address to prevent an attacker from performing this attack, in case any account was forgotten.

C.3 User Case Study #3

The participant had a Hotmail address that expired at an unknown date due to inactivity. The email address occurs in at least two database leaks; the Adobe leak and the Dropbox leak.

Through the password-reset function, we were able to get access to the Adobe account within a few minutes. In the Adobe account, we were able to see personal information such as address, phone number, and full name, as well as information on Adobe products the user downloaded or purchased. We were also able to get access to the Dropbox account, including access to personal account information, but more importantly access to all files stored, including personal files. Finally, Amazon provided us with two options to log into the user account, password reset, and a one-time login code. We found this second variant very dangerous, as it allows an attacker to stealthily login to the Amazon account and see personal information and the history of orders. In accordance with the study participant, we ended the study without actually logging in to Amazon and Dropbox after the password recovery. As in the other case studies, our recommendation to the study participant was to review all accounts to find references to the outdated email address and update it.

C.4 User Case Study #4

The participant had a Hotmail address that probably expired in 2011 due to inactivity. The email address occurs in at least two database leaks; the Myspace leak and the Exploit.in combination list.

Through the password-reset function, we were able to gain access to the actively used Facebook account. In accordance with the study participant we ended the case study at this point and ensured that the participant changes the reference from the expired email address to new one. We also recommended to review all accounts to find references to the outdated email address and update these references.

C.5 User Case Study #5

The participant had a GMX address that expired at an unknown date due to inactivity. The email address does not appear in any database leaks. However, we found that it was still referenced in the Amazon account. We found that by guessing the postal code, we were able to take over the Amazon account. In accordance with the study participant, we ended the case study at this point. As in the other case studies, our recommendation to the study participant was to review all accounts to find references to the outdated email address and update it.

C.6 User Case Study #6

The participant had a GMX address that expired at an unknown date due to inactivity. The email address is contained in several database leaks, including two commercial gaming platform accounts. We were able to recover access to both gaming platform accounts, allowing us to gain personal information. We provided the study participant with our recommendation to review all accounts and update any outdated email references.