USENIX

THE ADVANCED COMPUTING
SYSTEMS ASSOCIATION

Phishing Attacks against Password Manager Browser Extensions

Claudio Anliker, Daniele Lain, and Srdjan Capkun, *ETH Zurich*

https://www.usenix.org/conference/usenixsecurity25/presentation/anliker

This paper is included in the Proceedings of the
34th USENIX Security Symposium.

August 13–15, 2025 • Seattle, WA, USA

978-1-939133-52-6

Open access to the Proceedings of the
34th USENIX Security Symposium is sponsored by USENIX.

Phishing Attacks against Password Manager Browser Extensions

Claudio Anliker
ETH Zurich

Daniele Lain
ETH Zurich

Srdjan Čapkun
ETH Zurich

Abstract

We study a phishing attack against *password manager browser extensions*. Browser extension UIs are mostly displayed on top of the web browser's viewport and, thus, hard to distinguish from website content. This enables an attacker to phish master passwords by imitating a locked password manager on a website they control.

We implemented this attack for four password managers and demonstrated its effectiveness in a large-scale phishing simulation with 29,800 participants, among whom we detected over 400 instances of selected third-party password managers. Notably, more than 30% of these users entered their master password, with up to 58% for one specific password manager. We compare the effectiveness of the attack across different password manager UIs, analyze user behavior through mouse tracking and a post-study survey, and discuss the implications of our findings for password managers as a means of phishing protection.

1 Introduction

Password managers (PM) are widely used to generate, store, and autofill passwords and other credentials. These credentials are typically protected by a *master password*, which must be entered to unlock the contents of the *password vault*. A PM removes the burden of memorizing passwords, allowing users to choose stronger ones.

All modern browsers include a built-in PM, and many commercial vendors provide solutions with desktop and web applications, as well as browser extensions. PMs integrated into browsers are often advertised as a useful aid against *phishing*: since they only suggest or autofill credentials on associated websites, they can alert users to mismatched URLs and, consequently, most phishing attacks.

Given that PMs store many credentials, they are a valuable target for adversaries. While early PMs suffered from security issues [30, 44] and bad cryptographic practices [20], modern PMs are mostly victims of *conventional* phishing attacks [11], where users are directed to a counterfeit PM website via links in emails or malicious advertisements.

In contrast, we focus on **phishing via password manager browser extensions**. In this attack, the adversary mimics a locked PM by embedding a replica of the browser extension's login interface into a website they control. If the victim falls for the phishing attempt, they may enter their credentials into the adversary's website, in an effort to unlock their PM.

This type of attack was demonstrated in 2016 against LastPass in a proof-of-concept called *LostPass* [13], but it was not, to the best of our knowledge, investigated in academic research or exploited in the wild. Therefore, it is unclear whether PM users would fall for this attack or recognize the deception, especially in light of the continuous improvement of browser UIs over the last decade [40]. Furthermore, PM browser extensions differ in design and login workflows - factors that may affect their susceptibility to the attack.

In this work, we build on the observations from [13] and perform the first systematic, large-scale study of the susceptibility of PM browser extensions to phishing. Understanding the feasibility of this attack is important for two reasons: First, if it is successful, it grants the adversary access to all stored credentials, unless the PM is protected with phishing-resistant multi-factor authentication (MFA). Second, this attack differs from conventional phishing in the following ways:

1) Trusted UI confusion: The key to creating realistic impersonations of PM browser extensions lies in a design weakness of browser UIs: most trusted extension UI elements are rendered on top of the *viewport*, where the browser displays web content. This makes it difficult for users to distinguish their PM from a malicious copy embedded into an attacker-controlled website. Anecdotal evidence during the preparation of our user study suggests that the concept of a website impersonating browser extensions is surprising even to users with a technical background.

2) Diverting user attention: While browsing the web, users are likely to focus on the websites they visit rather than the PM, whose primary purpose is to facilitate login workflows. This benefits the adversary in two ways:

First, the adversary can induce the victim to unlock their PM by directing them to a legitimate-looking *secondary target*, a website or service that requires authentication. If the deception is successful, the victim will try to use their PM to get the password and, if the PM is locked, expect a prompt for their master password. If the adversary shows the spoofed PM at this moment, the victim is unlikely to suspect a phishing attack. In contrast, phishing that targets the PM's web login puts the PM into the center of the victim's attention.

Second, the attack becomes more flexible, as any website where the victim has an account can serve as a secondary target. Choosing a website perceived as low risk, such as a news site, can further reduce suspicion.

3) Attack from legitimate websites: Phishing attacks can also be executed via a compromised *legitimate* website. This type of attack is more difficult to detect, since the website's URL and other phishing indicators appear legitimate. Conventionally, an adversary could only harvest credentials for the affected website in this scenario. However, in the attack we study, they can exploit the victim's trust in the legitimate site to phish PM credentials, which greatly increases the threat of such hacks to end users.

Paper summary. To study the effectiveness of this attack, we conducted a large-scale phishing simulation involving 29,809 participants. The phishing email directed recipients to a *spoofed version* of our institution's SSO login portal, hosted on an external domain. On this website, we presented PM users with an impersonation of their browser extension UI to test whether they would enter their master password.

Our experiment showed a high average attack success rate of 31.25%, with a peak of 58.82% for one specific PM. The attack was consistently successful across various experimental conditions, from operating systems and browsers to different demographics, confirming that it poses a concrete threat.

Contributions. We make the following contributions:

- We show that this attack is applicable to *modern* web browsers, despite new trusted UI indicators, such as *extension tag indicators* or revised *extension popups*.
- We demonstrate in a large-scale real-world study that the attack is not only technically feasible but works in practice against different PMs on various browsers and systems, regardless of their design and how they display the login interface.
- We analyze user interactions and survey results to understand the behavior of PM users; in particular, we find that many of them prefer typing passwords they already know rather than unlocking their PM. Furthermore, users willing to use their PM are unlikely to be discouraged by imperfect UI impersonations, even if they notice visual discrepancies.
- We discuss potential countermeasures and highlight the challenges in solving the issues exploited by this attack.

2 Background

Web browsers GUIs. The Graphical User Interface (GUI) of a web browser window consists of the *browser chrome* and the *viewport*. The browser chrome is the trusted upper part of the window containing elements like the address bar, bookmarks, or browser extension icons. In contrast, the viewport renders web content and is, thus, completely under the control of the currently visited website. Within the browser GUI, a phishing attack can thus only be reliably detected using indicators in the browser chrome, such as the URL in the address bar.

2.1 Password Managers

Password managers (PM) generate, store, and provide passwords and other credentials. Stored credentials are encrypted with a secret key derived from a master secret (usually a password), which is required to access the contents of the *password vault*. While early PMs were stand-alone desktop applications [15], web browsers soon started to provide password management features[1]. Finally, a growing number of third-party vendors offer PMs as web browser *extensions* that fill in passwords on websites the user visits. Most of these extensions store passwords in the cloud to enable seamless synchronization between different devices.

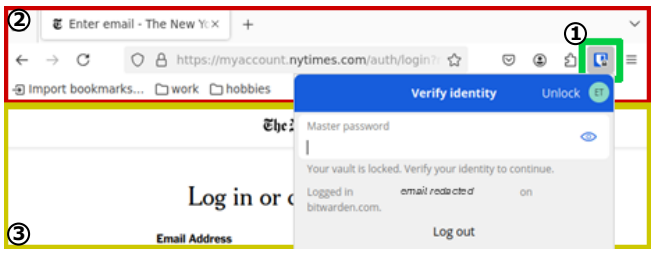
2.1.1 Web and Desktop Applications

Most PM vendors offer web and desktop applications with additional functionality compared to browser extensions, such as changing the master password, managing trusted devices, or adjusting advanced profile settings. However, these applications can be less convenient than browser extensions because they often require users to switch between programs to fill in passwords or save new ones. Depending on the PM, installing the desktop application may install the browser extension automatically (e.g., LastPass), change the extension's appearance (e.g., 1Password), or enable to unlock it via the system's authentication mechanism. However, system authentication does not completely replace the master password, which can still be required from time to time to unlock the vault or when using the PM on other devices.

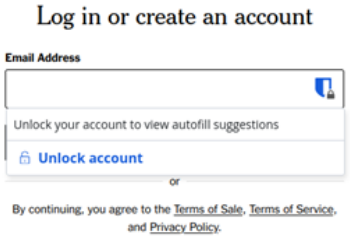
2.1.2 Browser Extensions

A PM browser extension is a trusted piece of software that integrates a third-party PM into a web browser. Compared to other web or desktop applications, browser extensions reduce user friction by highlighting password prompts and automatically filling in (or at least suggesting) passwords. Figure 1 shows common extension UI elements that are relevant for this work and discussed in the following.

[1] For example, Internet Explorer 7 since 2006, Google Chrome from its first version in 2008, and Firefox's predecessor Firebird as early as 2002.



(a) The *extension popup* is attached to the *extension icon* ① and is part of the *browser chrome* ②, the header of the browser UI. It overlaps the *viewport* ③ where the website is rendered.



(b) The Bitwarden *input decoration* comprising an icon and a panel with an interactive button ("Unlock account"). The input decoration is injected into the website's DOM.

Figure 1: **Common UI elements** of password manager browser extensions.

Extension icon and popup. PM browser extensions feature an *extension icon* in the browser chrome, which can be clicked to open the *extension popup* (or simply *popup*) of the password manager (see Figure 1a). When the PM is locked, the popup shows the login (or *unlock*) interface asking for the master password or other credentials. Otherwise, it displays stored passwords, settings, and profile information. Although the popup is part of the trusted browser chrome and cannot be opened or interacted with by websites, it is mostly rendered on top of the viewport.

Input decoration. PM browser extensions typically highlight login interfaces on websites and suggest credentials by injecting a small UI into the web form's HTML. This UI is often referred to as the *inline autofill menu* [1]. When locked, PMs typically replace the inline autofill menu with an icon indicating their presence and state; some also display a tooltip or small panel with unlock instructions (see Figure 1b). Since there seems to be no established terminology for these UI elements, we collectively refer to them as *input decoration*.

Unlike extension popups, which are part of the trusted browser UI, the input decoration consists of HTML code that is directly embedded into the Document Object Model (DOM) of the website. While browsers offer some protection against malicious modifications of these elements[2], we observe that it is still possible to detect and remove them. We will leverage this to build our attack.

Popup pages and application windows. While all PMs show a master password prompt in the extension popup, some of them also support browser popup pages or application windows, which we show in Figure 2.

A *browser popup page* or *popup page* (Figure 2a) is a new browser tab opened by the PM. The address bar displays a local URL, which typically contains the extension's UUID, making it rather opaque. In addition, most browsers also feature an *extension tag indicator* on the address bar, a graphical element bearing the extension's name, to emphasize that this is a local resource and not a website. LastPass and Dashlane are among the PMs that open a popup page when the user clicks on the input decoration. Note that while the URL and the extension tag indicator belong to the trusted UI, the unlock interface is completely rendered within the viewport.

An *application window* is a window opened by the operating system, with the OS-specific frame containing a title and icons, and a corresponding icon in the taskbar. Figure 2b shows the application window opened by default by Bitwarden; it has the same dimensions as the extension popup and is rendered in the same screen location, or even completely on top of the website-controlled viewport.

3 Motivation and Research Questions

3.1 PM Browser Extensions as a Target

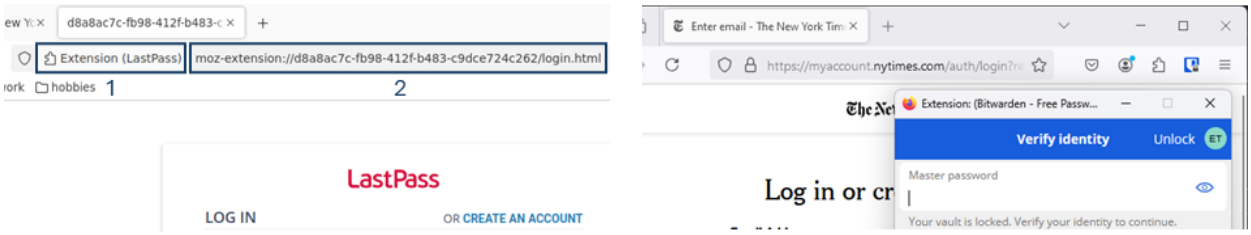
Phishing password managers (PMs) via counterfeit web logins is a form of conventional phishing, which has been extensively studied, and such attacks have been observed in the past [11]. In contrast, phishing PM browser extensions is interesting from a research perspective for several reasons:

Trusted UI confusion. As discussed in Section 2, PM browser extensions use extension popups, popup pages, or application windows to display the master password input field. Crucially, all of these components are (almost) fully rendered on top of or within the untrusted viewport. This is illustrated in Figure 3: the real browser extension popup (Figure 3b) can only be distinguished from the malicious copy (Figure 3a) by the toggled extension icon and an almost imperceptible overlap with the browser chrome [3]. We posit that these differences are too small for users to notice [31], which enables an adversary to trick them into leaking their master password.

Diverting user attention. We assume that the user's attention is primarily on the website they want to visit, and unlocking the PM is perceived as a necessary inconvenience. Furthermore, most users' knowledge of Internet safety revolves

[2]The injected element is inserted in a *shadow DOM* that isolates its content from the website and its Javascript code.

[3]The overlap is more pronounced in the presence of toolbars or bookmarks.



(a) LastPass *browser popup page*: the *extension tag indicator* ① and the URL ② are proof that this is a local website of the extension.

(b) Bitwarden *application window*: while it is possible to recreate it with HTML/CSS, the copy cannot leave the viewport.

Figure 2: **Password manager logins** in a browser popup page and an application window.

around the URL [6], not the subtle UI discrepancies discussed above. Finally, we assume that users check for phishing indicators, if they do so at all, before they decide to unlock their PM; if they trust the website and want to log in, they will likely also trust the PM popup we display, especially if they open it themselves by clicking on our injected icon in the website's login form. This is explained in more detail in Section 4.

Attacks from legitimate websites. An adversary can also launch this attack from a compromised service or website. In this case, all conventional phishing indicators would be legitimate, and even attentive and security-aware users could only spot the phishing attack based on the minor imperfections of the PM impersonation. Consequently, this attack increases the risks associated with a compromised website.

3.2 Research Questions

The above points make PM browser extensions an interesting target for phishing attacks, and we ask ourselves the following research questions:

RQ1: Does phishing of PM browser extensions work? Low-quality copies of websites and mobile apps are still sufficient to deceive users into entering their credentials [14, 31], and there is no obvious reason why a similar attack on browser extensions should not work. However, to the best of our knowledge, this has neither been studied by researchers nor exploited by cybercriminals, prompting the question if this attack is as effective as one might expect.

RQ2: Does PM design matter? While a counterfeit PM browser extension may look quite realistic, it cannot copy the original perfectly. For example, Figure 3 shows that the malicious version cannot toggle the extension button nor overlap the trusted browser UI. There are other details that are either impossible to fake, such as extension URLs in popup pages, or at least hard to know in practice, such as profile pictures. If these differences alert users to the attack, they could be starting points to improve PM user interfaces. Therefore, we ask ourselves whether PM designs influence the effectiveness of the attack.

RQ3: Which user groups are vulnerable? Finally, we ask ourselves whether any demographic groups are more susceptible to this type of attack, as this has been widely studied in the context of conventional (form-based) phishing, but less so in our scenario.

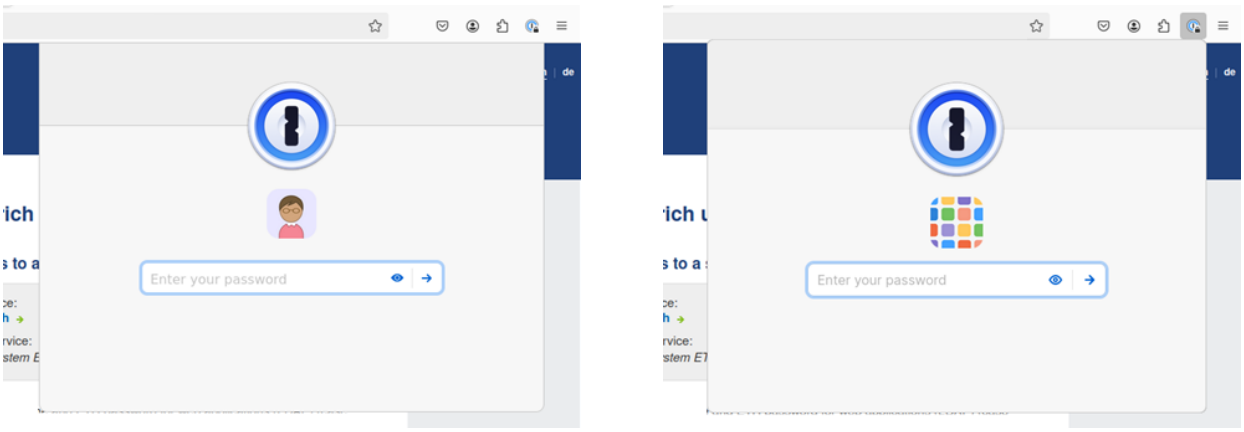
4 The Attack

Our attack exploits the fact that the trusted UI elements of a PM are hard to distinguish from an impersonation displayed on a website, as most of these elements overlap the attacker-controlled browser viewport (see Figure 1a and Figure 2). The attack uses a secondary target (e.g., a login or registration form on the mentioned website) as bait, which should motivate the victim to interact with the forged PM extension.

Threat Model. Our adversary either builds their own website or compromises a legitimate one. In either case, they inject the code that sets up the attack. Afterwards, the adversary lures the victim to this website through any means, e.g., email links. The malicious code is executed in the context of the victim's browser, which allows the adversary to access the DOM and JavaScript APIs. Once the victim interacts with the forged PM UI, and the adversary captures the master password, they can use it immediately to log in to the victim's PM account in a *runtime phishing* attack [46]: if the account is protected by multi-factor authentication, we assume the adversary to prompt the victim to provide the second factor as well. Finally, more detailed information on the victim might increase the chance of success. However, in our paper, we do not rely on any such information, and we do not target other authentication factors besides the master password.

4.1 Attack Steps and Technical Details

We assume the victim is lured to a phishing website and intends to log in. The attack then consists of (i) identifying the password manager; (ii) suppressing its input decoration; and (iii) displaying our own PM on the website.



(a) Forged popup. (b) Real popup.

Figure 3: **Forged and real extension popups for 1Password.** The forged popup does not feature the user's profile picture and is limited to the viewport; thus, it cannot slightly overlap the browser UI at the top edge like the real popup.

Step 1) Detecting the PM. PMs try to differentiate between logins and other web forms, since suggesting passwords in the wrong context could cause usability and privacy issues. We found that the heuristics to detect login fields differs between PMs and sometimes even between versions. In general, PMs check for the presence of input fields of type `password` (which hides typed characters) and HTML elements with keywords such as "username", "email", "password", or "pw" in their ID or name attributes.

We use this to both identify a PM and to suppress its input decoration. For the former, we add an invisible `password` field to the HTML code of our phishing website. When the website is being loaded, we install a `MutationObserver`, made available via the browser's JavaScript API, which enables us to monitor the entire DOM for modifications. Once the page has been loaded, the PM of the user will detect our password field and decorate the login with an icon or similar elements. This change triggers the handler function of our observer, which receives the injected node as a parameter. Browser extensions typically wrap the input decoration in a *shadow DOM* to isolate it and prevent inspection by code running in the website. However, the root node containing the shadow DOM can be accessed, and we check the tag, name, and attribute of this node against a list of known modifications: for 1Password, we look for HTML nodes with "1password" in their tag name, such as `<com-1password-button>`, for LastPass, Bitwarden and Dashlane, we check HTML attributes of injected nodes whose names contain the corresponding PM name (e.g., "data-lastpass-icon-root") or are otherwise unique for a given PM.

Once we detect a PM, we delete the invisible password field and with it all injected elements. While the input decoration might be visible for a split second in rare cases, the entire process usually runs too fast to be undetectable for the user.

Step 2) Suppressing the PM. Simultaneously, we need to prevent the real PM from adding its input decoration to the actual login form of our phishing website, as this would interfere with our impersonation. Based on our understanding of how PMs detect logins (see Step 1), we disguise ours as an ordinary web form by using different input types and field names, removing attributes, and rectifying the resulting visual changes with manual CSS code.

Step 3) Forging the PM. Every PM impersonation consists of two HTML files for the input decoration and the PM login, respectively. Both of them are isolated in shadow DOMs to prevent style leaks into the website and vice versa. For every PM, we also add a JavaScript class containing its functionality (i.e., setup routines and event handlers). All files are already embedded into our phishing website from the beginning, but are hidden from view.

After Step 1, we show only the input decoration of the PM we detected and activate its event handlers, for example to display the PM login when the user clicks on the input decoration. For the purpose of this study, we added further code to track mouse events, which are aggregated on the client and sent to our server at set intervals, whenever the user enters a password, or when they close the tab.

All PM impersonations follow the same workflow, with one notable exception: for some LastPass users (see group L_{Page} in Table 2), we simulate a browser popup page by displaying the PM login in a new browser tab. In this case, the details depend on the browser, as they all represent local resources slightly differently. For example, Mozilla Firefox loads the LastPass login as `moz-extension://<UUID>/login.html`, which we imitate with a URL of the form `https://moz-extension.<UUID>.<TLD>/login.html`. This deceptive domain points to the same web server hosting our phishing website.

4.2 Limitations of PM Forgeries

As discussed in Section 6.2, certain limitations prevent the adversary from perfectly impersonating PM browser extensions. We briefly outline these limitations and assess their potential impact on the success of the attack.

Extension popup. The extension popup is attached to the extension icon in the browser chrome and overlaps the remainder of the chrome down to the top edge of the viewport, such as bookmarks or other toolbars (see Figure 1a). Without such elements, the unforgeable overlap is almost imperceptible (see Figure 3b). However, another clue is the password manager's extension icon, which is only active or pressed (indicated by a change in background color) when the real popup is shown (c.f. Figure 3a and Figure 3b).

Browser popup page. Two elements are not forgeable by an adversary: the extension tag indicator, showing the extension's name, and its local URL (see Figure 2a). Instead, an adversary forging this type of UI needs to open a new tab and load an external website, whose address will be shown in the URL bar instead. However, this difference might be too small or too difficult to detect, given users' struggles with URLs [39] and the opaque nature of the strings extensions display as URLs on their pages (see Figure 2b).

Application window. There are two approaches to impersonating an application window like the one shown in Figure 2b. The first is to open a popup using the browser API to load an external resource. However, such a popup contains an address bar showing the resource's URL. This address bar cannot be removed—a security measure to prevent the kind of UI confusion this attack relies on. Since the address bar is the only clue, it stands to reason that an otherwise perfect UI might still deceive some users.

Alternatively, the adversary can replicate the application window within the website (similarly to the extension popup) by crafting the frame and adding the icons of the application window manually. This allows a pixel-perfect forgery, and we follow this approach in our phishing simulation for one group of Bitwarden users. The limitation of such a window is that it is restricted to the viewport, and interactions like dragging it outside of the tab or maximizing it to full screen will not work. Additionally, it will lack the icon that operating systems typically add to the taskbar when the window is created.

Custom UI components. Some UI elements are customized according to the victim's system configuration and cannot be replicated exactly. However, unlike the previously discussed limitations, which an adversary cannot overcome, these elements can potentially be guessed. For example, 1Password displays the account's profile picture in the login interface, and other PMs show the email address. Finally, the language reported by the browser's User Agent might differ from the language used in the system, causing the adversary to serve deceptive content in the wrong language.

5 Study Design

To answer our research questions on the feasibility and performance of our attack, we conducted a large-scale phishing simulation with 29,808 participants at an institution of higher education. Our simulation (which was not announced, as it is customary in such studies to avoid priming participants [19]) featured a convincing and sophisticated phishing email and a phishing website mimicking the institution's Single Sign-On (SSO) portal, sent from a typo-squatted domain resembling the original (further details in Section A.1). Here, participants were presented with an impersonation of their PM browser extension, assuming they used one of the PMs we tested for. Participants without such a PM where only shown the SSO login.

We recorded how participants interacted with the website, and whether they entered data either into the SSO login interface or into the spoofed PM extension UI. Participants were then debriefed and invited to fill in an optional questionnaire.

Mobile devices. We expected many participants to read the email and visit the phishing website on a smartphone or a tablet. Since PMs work differently on these devices, they were not in the scope of our study. We tried to redirect these participants to laptop and desktop computers, using the pretense that the website does not yet support mobile devices.

5.1 Supported Password Managers

Our study targeted four popular PMs: Bitwarden, 1Password, LastPass, and Dashlane. All of them open the login in an extension popup when the user clicks the extension icon in the browser chrome, but they handle clicks on the input decoration differently: Bitwarden opens an application window, LastPass and Dashlane open popup page, and 1Password's input decoration ignores clicks completely[4], inviting users to click the extension icon in the browser UI instead.

These differences may affect users' expectations of how their PMs behave, and thus affect the success rate of the attack. To estimate these effects, we tested several configurations, as shown in Table 1: for Bitwarden, we tested if users are more susceptible to a PM impersonation in an application window (group $\mathbf{B}_{AW,Win}$) or an extension popup (group $\mathbf{B}_{Ext}$). For 1Password, we provide a Safari-specific UI, a version with a default profile picture, and one with no picture at all. For LastPass, we render the PM login either in an extension popup or a browser popup page.

5.2 Data Collection

We collected three types of data from participants: whether they entered any credentials into either the SSO or the PM

[4] If the 1Password system application is installed, clicking the input decoration shows its login window.

Table 1: **User groups**. Participants with a detected PM were divided into the following groups, depending on their configuration.

Password Manager	Group Name	Details
Bitwarden	$\mathbf{B}_{AW,Win}$	PM login in an application window (see Figure 2b); Windows users only.
	$\mathbf{B}_{Ext}$	PM login in extension popup (see Figure 1a); the sub-group of Windows users is $B_{Ext,Win}$.
1Password	$\mathbf{1P}_{Ext}$	Extension popup, default placeholder profile picture.
	$\mathbf{1P}_{Ext,NoPic}$	Extension popup, no picture.
	$\mathbf{1P}_{Safari}$	Extension popup matching the look of the Safari extension.
LastPass	$\mathbf{L}_{Ext}$	Extension popup.
	$\mathbf{L}_{Page}$	Browser popup page (see Figure 2a).
Dashlane	$\mathbf{D}_{Ext}$	Extension popup (real extension uses a popup page instead).

login interfaces, how they interacted with the website (e.g., where they clicked, whether their cursor left the browser viewport, and when the browser was closed) and their answers to the optional questionnaire.

Password inputs. Due to ethical and data protection reasons, we neither recorded the SSO passwords nor the PM master password in our study.

To understand whether participants entered their password in either login form, we leveraged password metadata. First, we tried to elicit two password inputs by always showing an error that the first one was incorrect. We then compared the two inputs and only recorded their lengths and their Levenshtein distance. Our decision heuristic for genuine login attempts was as follows:

SSO password: (i) at least one password input is at least 12 characters long, matching the institution's password policy; (ii) the Levenshtein distance between the two inputs is at most 2; and (iii) the Levenshtein distance between the username input and the real username is at most 2.

Master password: (i) the minimum length of password inputs is 10 characters; (ii) their Levenshtein distance is at most 2.

We relaxed the length requirement for the PM master password because some PMs had weak password policies in the past (e.g., Bitwarden [8]) and did not force users to update their master password.

Interactions. We added listeners to several UI elements of the SSO login interface, the input decoration, and the PM login to track the `mouseenter`, `mouseleave`, and `click` events, including cursor positions and timestamps. The events were aggregated on the client side and sent to the server in batches until the participant closed the tab or the session expired.

5.3 Debriefing

Due to the deceptive nature of the study [19] and to mitigate potential distress, we debriefed all participants as soon as possible. Our debriefing comprised a debriefing email and a debriefing website. The debriefing email (see Section A.2) provided a self-contained debriefing with information we deemed necessary for the average user. The email also pointed to the debriefing website, which contained an additional Q&A with answers to common questions.

Our debriefing strategy was as follows:

- When a user visited the website for the first time, we scheduled the debriefing 20 minutes later. If the user stayed on the website that long, they were automatically redirected to the debriefing page, and the debriefing email was sent simultaneously.
- If a participant made one login attempt with either the SSO or the PM login form, the debriefing was rescheduled to happen in five minutes instead. The reasoning behind this decision was to reduce the period of potential anxiety for participants who detect the attack after entering *one* password, while still giving those who get distracted between inputs a chance to finish the experiment.
- Participants who made two login attempts using either form were immediately redirected to the debriefing page, and the email was sent simultaneously.
- If a participant did not visit the phishing website, they received a slightly modified debriefing email at the end of the simulation (after less than one week).

6 Results

We sent phishing emails to 29,809 participants; 12,006 (40.28%) visited the phishing website, and 9,422 (31.61%) did so on a supported device, namely a laptop or desktop computer. After the simulation, 3,547 participants filled in the survey.

Overview. Table 2 provides an overview of our results: in total, 6,257 participants entered their SSO password, which corresponds to 66.41% of visitors with supported devices and 20.99% of all participants. We refer to the success probabili-

Table 2: **Performance of participants that used a supported password manager**. For each group, we report how many fell for the PM browser extension phishing and how many for the SSO login phishing. Note: group $\mathbf{B}_{Ext,Win}$ is a subset of $\mathbf{B}_{Ext}$.

PM	Group	Detected	Master password		SSO password		Combined	
		#	#	of det.	#	of det.	#	of det.
Bitwarden	$\mathbf{B}_{Ext}$	172	46	26.74%	47	27.33%	91	52.91%
	$\mathbf{B}_{Ext,Win}$	57	14	24.56%	22	38.60%	35	61.40%
	$\mathbf{B}_{AW,Win}$	47	9	19.15%	21	44.68%	30	63.83%
	all	*219*	*55*	*25.11%*	*68*	*31.05%*	*121*	*55.25%*
LastPass	$\mathbf{L}_{Ext}$	27	0	0.0%	16	59.26%	16	59.26%
	$\mathbf{L}_{Page}$	30	3	10.00%	15	50.0%	18	60.00%
	all	*57*	*3*	*5.26%*	*31*	*54.39%*	*34*	*59.65%*
1Password	$\mathbf{1P}_{Ext}$	44	19	43.18%	15	34.09%	32	72.73%
	$\mathbf{1P}_{NoPic}$	50	20	40.00%	18	36.00%	38	76.00%
	$\mathbf{1P}_{Safari}$	44	23	52.27%	12	27.27%	34	77.27%
	all	*138*	*62*	*44.93%*	*45*	*32.61%*	*104*	*75.36%*
Dashlane	$\mathbf{D}_{Ext}$	34	20	58.82%	7	20.59%	27	79.41%
All PMs	-	**448**	**140**	**31.25%**	**151**	**33.71%**	**286**	**63.84%**
No PM	-	8,974	-	-	6,106	68.04%	6,106	68.04%
All	-	9,422	140	1.49%	6,257	66.41%	6,392	67.84%

ties of the SSO phishing and the PM phishing as *SSO success rate* and *PM success rate*, respectively.

We detected our target PMs in the browsers of 448 participants, whom we refer to as *PM users*. Bitwarden was the most common PM (219 unique visitors), followed by 1Password (138), Lastpass (57), and Dashlane (34). In total, 140/448 PM users (31.25%) entered the master password and 151 (33.71%) the SSO password. Finally, 144 PM users submitted the survey.

We first summarize our main findings before discussing them in more detail in Section 6.1 - Section 6.3.

RQ1: Does phishing of PM browser extensions work? The PM success rate of 31.25% proves that the attack poses a concrete threat. We also observe that many users prefer to type in a known password manually rather than unlock their PM, and we conclude that the choice of the secondary target has a considerable effect on the attack.

RQ2: Does password manager design matter? Our results show that rendering the PM login form in an extension popup, rather than in an application window or a popup page, does not reduce the PM success rate. The same applies to 1Password's profile picture; replacing it with a default avatar or removing it altogether does not notably reduce the attack's success. We further found that many users entered their master password despite noticing visual discrepancies in the PM UI. Based on these observations, we expect that UI changes within the capabilities of a browser extension can only play a minor role in mitigating the attack.

RQ3: Which user groups are vulnerable? Our results do not show notable differences between participants from different backgrounds. In particular, the attack was similarly successful against participants affiliated with IT, computer science, and electrical engineering, contrasting prior results for traditional phishing [21]. Within the limitations of our study population (see Section 7.2), our results further suggest that participants are vulnerable across all age groups and independently of their OS or browser.

Methods. Whenever we refer to a result as (statistically) significant, we verified it with a Fisher's exact test (unless specified otherwise). We consider a result significant if $p < 0.05$, even if p is not reported explicitly for editorial reasons.

6.1 RQ1: Phishing the PM Browser Extension

Table 2 shows that the attack had a success rate of 31.25% on average, and even 44.93% and 58.82% for 1Password and Dashlane, respectively. In contrast, the attack was less successful against LastPass users, with only 3 of 57 participants (5.26%) entering their master password. We discuss this result in detail in Section 6.1.1.

F1: PM browser extension phishing is a real threat; on average, 31.25% of participants entered their master password.

Detectability of the attack. 151 of 448 PM users (33.71%) entered their SSO password but not their master password. We posit that these participants did not detect the PM phishing, as they would have left the website otherwise. We suspect that most of them either copied the SSO password from their real PM or entered it manually, and we analyze this further below.

On average, the combined success rate of the SSO phishing and the PM phishing is 63.84% for PM users, which is similar to that of non-PM users (68.04%), suggesting that our attack was largely inconspicuous. For the subgroup of 335 PM users who opened our PM login form (by clicking on the input decoration), the combined success rate reaches 73.13%, surpassing non-PM users. This further shows that most participants did not detect our attack.

PM users entering the SSO password. We analyzed our survey data to investigate why so many PM users entered their SSO password directly. The corresponding questions Q1, Q2, Q3, Q11, and Q12 can be found in Section A.3.

We start with participants who indicated, in response to the following question, that they use a PM:

Q2: Please select the password manager you use the most for storing website passwords.

The answers are illustrated in Figure 8 in the appendix[5]: in total, 2,202 of 3,457 participants reported using a PM, with the majority relying on the one integrated into their web browser, followed by the iOS/macOS Passwords app and KeePass. Figure 8b shows that we chose four of the five most popular browser extensions in our population. Of the 2,202 self-declared PM users, 1,800 (81.74%) store their SSO password in the PM mentioned (Q3), but 1,442 of these 1,800 (80.11%) also stated that they know it by heart (Q1). In the end, 547 of the 1,800 participants who store their SSO password in the PM admitted entering a password (Q11), but only 211 (38.57%) said they had used their PM in the process (Q12).

This data suggests that the majority of self-declared PM users did not need their PM to enter their SSO password. Apparently, many of them followed their original intention of authenticating to the SSO portal, not bothering with their locked PM first.

6.1.1 LastPass

In this section, we explore three factors that may have contributed to LastPass' low PM success rate. We analyze the data of the 448 PM users, split into a LastPass group (57 participants) and a complement group (391 participants). For the survey, we only consider the 144 submissions pertaining to these groups, i.e., 23 from LastPass users and 121 from the complement group.

Was the attack on LastPass easier to detect? Our data in Figure 4 shows that 46 of 57 LastPass users, or 80.70%, did not click on our input decoration (which consists only of the gray LastPass icon, see Figure 7b) in the SSO login form. Therefore, they never saw our PM login, and detecting the attack based on the icon only is virtually impossible. Additionally, 54.39% of the LastPass population entered the SSO password, which is more than 21% higher than for any other PM and further indication that most of them did not detect the attack. This conclusion is also supported by the survey:

Q13: Regarding your password manager, did you notice anything out of the ordinary on the website?

This was a free-text question, and we grouped the answers into UI-related remarks and others: no one of the 7 LastPass users who answered the question mentioned the UI. In the complement group, 37 of 60 responses mentioned minor UI discrepancies related to colors, fonts, or positioning.

Did LastPass users open their real PM instead? It is not trivial to track accurately how many participants used the real PM, as this interaction happens outside of the browser tab we control. However, we can provide a rough estimate by counting how many participants left the viewport with their cursor, via the top edge, within 30 seconds before entering the SSO password. The result is likely to be an upper bound with false positives (e.g., users opening a bookmark, switching tabs, or scrutinizing the URL) outnumbering false negatives (i.e., users moving the cursor indirectly to their PM, not via the top edge).

We did not find a notable difference between the two groups: for LastPass, 11 of 31 LastPass users (35.48%) left the viewport shortly before entering their SSO password, compared to 43 of 120 (35.83%) of the complement group. The following survey questions corroborate this result:

Q12: On the website, did you use or attempt to use your password manager to enter your <redacted> password?

Only 14 LastPass users entered the SSO password and completed the survey (3 "*Yes*", 5 "*No*", and 6 "*I'd rather not say*"), compared to 82 users in the complement group (57 "*Yes*", 16 "*No*", and 9 "*I'd rather not say*"). Despite the small sample size, this results suggests that LastPass users were even significantly less likely to answer "*Yes*" ($p = 0.025$, excluding "*I'd rather not say*" responses).

Q8: If you visit a website and your password manager is locked, what is your first step? I start by ...

This was a multiple-choice question. For both groups, the most popular choices were "*clicking the icon in my browser*

[5]The sum of the numbers in Figure 8 is slightly higher because some participants mentioned more than one PM in the free-text option.

toolbar.", which would correspond to opening the real PM, and "*clicking the overlay/icon of my manager in the username/password field on the website.*", which would open our impersonation. We could not find a significant difference between LastPass users and the complement group.

LastPass users had to enter the email address. While we had access to our participants' institutional email addresses, we did not know the private ones they presumably used to create their PM accounts. For most PMs, this was only a minor inconvenience: 1Password only shows the email address when hovering over the profile picture, and Bitwarden and Dashlane display it in an inconspicuous, read-only field. For all three, we either substituted the user name or simply removed the email address (see Figure 7).

In LastPass however, the email address is a writable field and quite prominent (see Figure 7d). We decided that leaving the field blank would be less suspicious than removing it altogether, meaning the users had to enter it manually. This may have discouraged some of them from entering their credentials. However, this only applies to the eight people in Figure 4 who opened our LastPass login without entering their master password - and since six of them provided their SSO password instead, we conclude that the missing email was rather an inconvenience than a red flag for our users.

While the three factors above may have had some effect on the success rate of our attack, they fail to explain the outcome completely. We conclude that many LastPass users in our population were not susceptible to our attack because they preferred to enter their SSO password manually instead of using their PM.

6.2 RQ2: Effect of UI Elements

We wanted to know how attentive users are to personalized UI elements, and we studied this on the example of 1Password's profile picture. While a dedicated attacker might know this picture and, thus, impersonate the 1Password UI more accurately, we did not have access to this information. In addition, we analyzed whether the use of popup pages or application windows affects the success rate of our attack. Finally, we wanted to know if users detected any UI discrepancies in the UI forgeries, and how this affected their decision to enter their master password.

Profile picture (1Password). The browser extension of 1Password displays the user's profile picture on all systems, with the exception of "1Password 7" on Safari. Assuming the user does not use the default avatar, this picture is the only UI element of all tested PMs that is unknown at the time of the attack.

In our simulation, we showed users either the default avatar (group $\mathbf{1P}_{Ext}$) or no picture at all ($\mathbf{1P}_{Ext,NoPic}$). We observe from Table 2 that, surprisingly, the PM success rate is over 40% in both cases, even though either UI was different from what users were accustomed to (barring those participants who never changed their profile picture). This confirms previous findings that users tend to trust UIs even if their appearance deviates considerably from the originals [31].

Browser popup page. We found that the benefit of browser popup pages is negligible. Both Dashlane and LastPass open the PM login in a browser popup page by default. In our study, we changed this for all Dashlane users and for LastPass users in group $\mathbf{L}_{Ext}$ by rendering the PM login in an extension popup instead. We observed that most Dashlane users entered their master password regardless (see Table 2). They seem to have trusted the login despite the "wrong" location, potentially because they are accustomed to using the extension popup, too. Therefore, we conclude that popup pages do not help to prevent PM phishing, as they can just be replaced by an extension popup in the attack.

Application window. Bitwarden typically displays the login in an application window. We wanted to know if changing the login's location had an effect on the attack's success and analyzed this by splitting our Bitwarden users on Windows into two groups: group $\mathbf{B}_{AW,Win}$ was shown an impersonated application window and $\mathbf{B}_{Ext,Win}$ an extension popup. Surprisingly, we found that showing an extension popup rendered the attack more successful, with a PM success rate of 24.1% (versus 19.6% for the application window). We conclude that there is no benefit in forging OS-specific application windows, in accordance of what we observed for Dashlane and the popup page.

Self-reported UI discrepancies. To see how participants perceived our UI impersonation, we revisit the following survey question:

Q13: Regarding your password manager, did you notice anything out of the ordinary on the website?

Among the 67 PM users who answered the question, 37 made remarks related to their PM UI, including the following examples, which may have been translated into English or modified minimally for editorial reasons:

"*The PW manager looked a bit strange, I assumed they had changed the design.*"

"*I noticed the UI and font of the password manager on the phishing website was different from the one I usually have.*"

"*Yes, the password manager had rounded corners and was not displayed in its usual position.*"

"*The colours seemed off but I did not think about it.*"

Surprisingly, 27 of the mentioned 37 participants (72.97%) still entered their master password, including all those who made the comments above, reinforcing previous findings that users tend to notice UI discrepancies but dismiss them [31, 41].

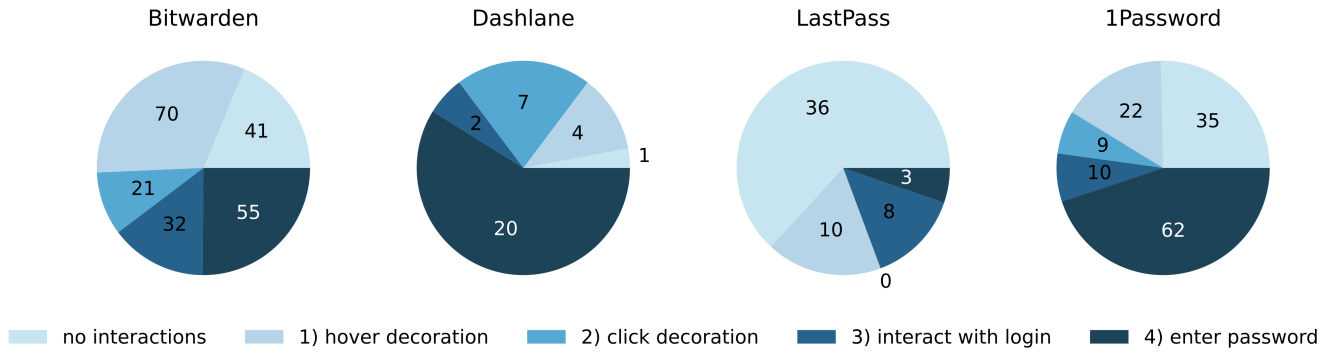


Figure 4: **PM user interactions with the PM impersonation.** To enter the master password, users had to perform actions (1) - (4). The plots show how many participants abandoned the process at each step. LastPass is discussed in Section 6.1.1.

> F2: Even major discrepancies in PM UI or behavior compared to the legitimate extension did not notably alert users towards the deception.

6.3 RQ3: Vulnerable User Groups

Finally, we investigated whether specific user groups are particularly vulnerable to PM phishing. Thus, we grouped our study population based on background, browser language, web browser, and OS.

Background. Since we conducted the study at an educational institution, our population comprises dozens of study programs and organizational entities and is highly diverse. We coarsely divided our participants into the following four groups: IT/CS/EE (IT Services, Computer Science, and Electrical Engineering), Math/NS (Mathematics and Natural Sciences), Admin (Administration), and Other (a range of mainly non-CS-related engineering departments).

Figure 5 shows that the PM success rate was similar for all four groups, with 32.00% for Admin (16/50), 26.80% for Math/NS (26/97), 28.65% for IT/CS/EE (49/171), and 37.69% for Other (49/130). Interestingly, IT/CS/EE performed similarly to the rest, contrasting with previous findings on traditional phishing [21]. Among Dashlane and 1Password users, susceptibility was even higher, indicating that PM phishing can also affect users with high technical expertise.

Age. Table 3 shows that all five age groups exhibit a PM success rate of between 28.57% and 34.52%. In terms of general feasibility, this demonstrates that the attack works similarly well against all age groups. Further, a Chi-squared test shows that these differences are not significant ($p = 0.879$). However, we note that the group under 30 years of age is considerably larger than the others, which is a consequence of conducting the study at a university: running the study with a larger, more evenly distributed population might produce different results.

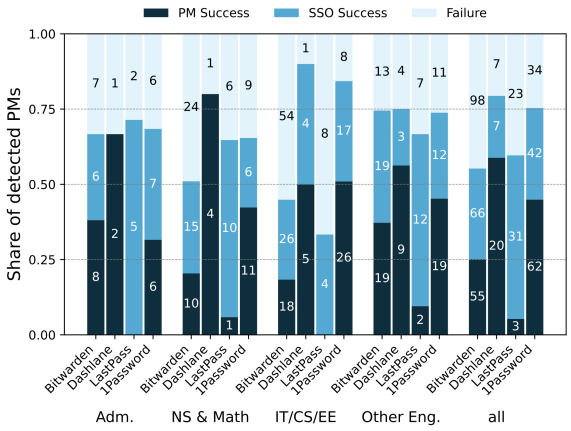
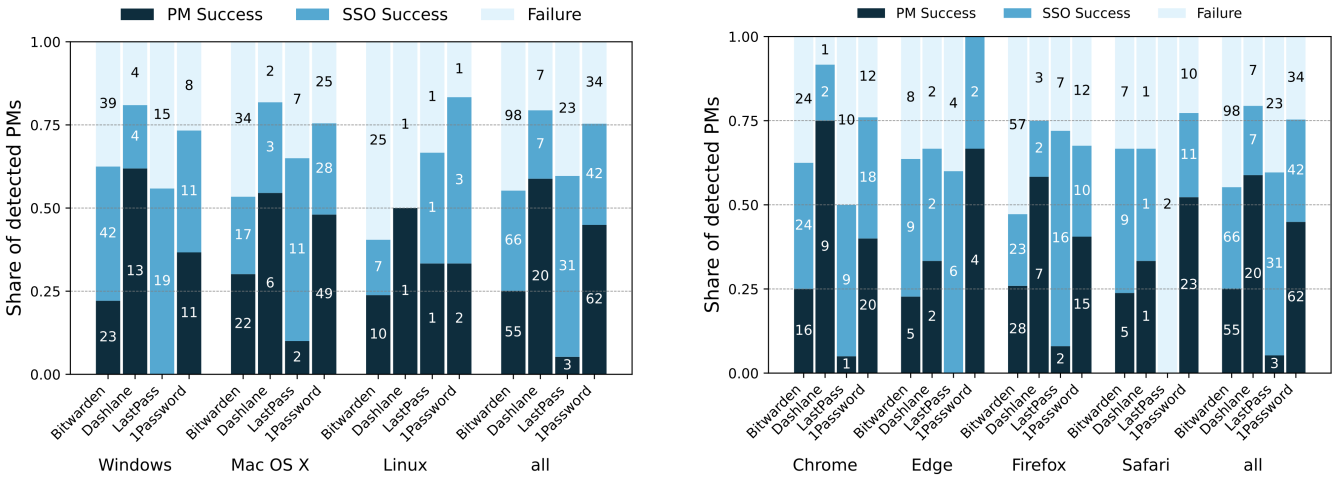


Figure 5: **Relative shares of PMs**, grouped by background.

Languages. Our PM impersonations were only available in German and English—the correspondence languages of our institution. The language was chosen at runtime based on the language preferences of the user's browser (i.e., the HTTP header `Accept-Language`). We found that these choices were sufficient; users with other browser languages formed a minority (8.68%), even more so among those with PMs (2.68%). While it is possible that the language of a user's PM does not match the browser preference, we assume this to be the exception. None of the detected PM users who filled in the survey made remarks related to their PM's language, and we could not detect a significant difference between groups interacting with German or English PM forgeries.

Browser & OS. Figure 6 shows the PM success rate for different OSs and browsers. For Bitwarden and 1Password, for which we have more data, we observe that phishing rates are consistent across all popular OSes[6].

[6]Note that there are only a few entries in some categories such as Dashlane or LastPass on Linux.



(a) PMs by Operating System.

(b) PMs by browser.

Figure 6: **Shares of detected PMs**, grouped by OS and browser.

7 Discussion

7.1 Countermeasures

We now discuss possible countermeasures to this attack from the perspective of the two main issues it exploits: (i) the high risk associated with leaking the master password; and (ii) the UI deception we described in Section 4.

7.1.1 Impact of Stolen Master Password

Two-factor authentication. A widespread countermeasure against the use of stolen credentials is two-factor authentication (2FA). Indeed, all studied password managers encourage OTP-based or approval-based 2FA for user logins [3,9,16,29]; and most perform device authentication on first access by confirming logins from new devices, e.g., by clicking on a link sent via email [17] [4]. However, these solutions are not phishing-resistant as they remain vulnerable to *runtime phishing* [46], in which the adversary also coerces the second factor from the victim, or convinces them to approve the new device. The same considerations apply to dedicated long-term credentials such as 1Password's *Secret Key*, which users provide only for critical security operations such as approving a new device login: while the hope is that users pay more attention to such credentials, successful phishing attacks on cryptocurrency wallet seed phrases [18] (despite users being instructed not to reveal them to anyone) suggest that they are equally vulnerable.

Further solutions in this space progressively trade off usability for security; however, while they raise the bar for adversaries, we observe that they do not offer complete phishing protection. One example is requiring users to click on a link, sent by email, on the specific device they want to approve: while this prevents the user from accidentally approving the adversary's login, the adversary can still trick them into forwarding such email. Phishing-resistant 2FA, such as those based on FIDO2 and hardware tokens [23] or proximity verification [24] prevent adversaries from stealing the second factor. However, PMs ultimately need to allow for account recovery (e.g., in case of a lost hardware token), making them vulnerable to phishing via *downgrade attacks* [45].

Moving forward from traditional credentials. Ultimately, we observe that there is an inherent vicious circle in which PMs that rely on master passwords are susceptible to the very same attack they aim to prevent. Modern approaches try to solve this problem by replacing passwords altogether: passkeys are a more secure alternative based on mutual cryptographic authentication between the client and server, which is also being adopted by password managers [2,10,28]. However, passkeys are unlikely to completely replace passwords in the near future [27]. Moreover, they share a fundamental limitation of other phishing-resistant authentication factors: the need to support account recovery through "conventional" means, which can be phished by an adversary.

7.1.2 UI Improvements

The key observation motivating this attack is that UIs of PM browser extensions can be impersonated almost perfectly. In the following, we address the main reasons for this issue and discuss potential countermeasures.

Confusion between trusted and untrusted UI. In 2010, the W3C issued a recommendation to improve the distinction between trusted and untrusted elements, stating "*Web User Agents MUST NOT communicate positive trust information using user interface elements which can be mimicked within*

chrome under the control of web content" [40]. Although our attack does not technically mimic a component *within* the chrome (the impersonated UIs are still in the viewport), the PM's input decoration and extension popup clearly convey trust, and current browser designs seem to violate the spirit of this recommendation.

However, resolving these issues is hard: removing input decorations, or making them non-interactive, would mean that users had to open their PM via the browser chrome. While this might train some users to ignore our UIs and prevent this particular attack, it is unclear how they would react if the PM impersonation were displayed automatically. More importantly, our study has shown that many 1Password users clicked on our input decoration even though this feature is not even available on 1Password on most browsers[7].

To prevent the intermingled rendering of extension UIs and website content, one starting point could be to fully move the PM UI into the chrome, i.e., the header part of the browser window. Rendering trusted UIs on untrusted background is a challenging problem (see past failed efforts by operating system dialogs such as user account privilege prompts in Windows [12]). We observe that this approach would be easier to realize in a browser (where the part the adversary can control is limited to the viewport) than in an operating system (where the adversary can control the entire screen). However, this may affect usability, since the space in the browser's chrome is quite limited. Furthermore, past efforts have shown that users are often not alerted by UI discrepancies in phishing attacks [31, 41], and our data confirms that most users who reported irregularities in the UI entered their master password anyway (see Section 6.2). Indeed, our study confirms that the attack works even when showing participants a PM login in a different position, suggesting that even a dedicated region in the browser chrome might not prevent users from entering their master password into a similar UI shown in the viewport.

Personalized UI elements. For the same reason, we reflect that adding personalized UI elements is not a reliable countermeasure. Past efforts have shown that users were not alerted by the absence of their profile picture [41]. Our study validates that this still holds almost 20 years later: 1Password users entered their password regardless of whether the UI contained the default avatar or none at all (see groups $\mathbf{1P}_{Ext}$ and $\mathbf{1P}_{NoPic}$ in Table 2).

Security Indicators. Finally, we observe that several indicators that users rely on to spot phishing are not present or not helpful in UIs of PM browser extensions. For example, while users often rely on URLs to assess the provenance of a website, i.e., the content they are seeing in the viewport of a website (albeit with limitations [5]), PMs rarely have such an indicator—when present, it is an opaque and complex URL[8] that does not provide sufficient help [5]. Other indicators such as the extension tag indicator (see Figure 2a) are encountered infrequently by users and their absence is a minor mismatch that is likely to go unnoticed [41]. Furthermore, these indicators only apply to extension popup pages: for extension popups, no dedicated indicator exists, and users can only rely on the aforementioned visual discrepancies (e.g., the extension button not being highlighted) to spot the forgery.

7.2 Study Validity

Word-of-mouth. Our phishing simulation ran at a university on two consecutive workdays during the academic semester, and students comprised the majority of the study population. In such an environment, information about the simulation was bound to spread through word-of-mouth and other communication channels, which likely prevented large numbers of participants from interacting with the phishing email and website.

University SSO as a secondary target. We used a copy of a university SSO portal as our secondary target, i.e., the phishing website to lure participants into interacting with the impersonated PMs. Our results indicate that most PM users knew their SSO password by heart, likely due to frequent use. This suggests that the choice of the phishing website (i.e., the secondary target) can substantially influence the success rate of this type of attack.

Generality of PM choice. We validated both the generality of our PM choice and the reliability of our PM fingerprinting technique by analyzing the answers to the post-study survey. Figure 8 shows the responses to the question "*Please select the password manager you use most frequently to store your website passwords.*". The majority uses a browser-native PM [32], the iOS/macOS Password app (or Apple Keychain), or no PM at all. However, 472 use a PM we supported in our study, and our choice reflects four among the five most popular third-party PMs with browser extensions; moreover, the reported PM shares roughly match our results, confirming that our detection worked similarly well for all PMs.

Practical limitations of UI impersonations. As discussed in Section 4.2, several PMs feature elements that an adversary can only guess, e.g., the profile picture of 1Password. In our study, we did not have this information and had to compromise, e.g., by showing the default profile picture. Furthermore, PMs usually display the email address of the connected account, and we only had the institutional emails of participants, not the private ones they likely use for their PM account. Finally, some PMs exist in multiple versions for different systems, often with subtle differences in appearance and interactions, and are subject to frequent updates. Consequently,

[7] On Windows and Linux systems, the icon is only clickable if the 1Password desktop application is installed.

[8] Such as `chrome-extension://...` or `moz-extension://...`.

it was not practical to tailor PM UIs for every system configuration. However, we believe that these limitations did not considerably alter the results: we saw in Section 6.2 that even intrusive UI modifications did not deter users from entering their master password.

Population. Our population, coming from an educational institution, is skewed toward participants with higher education and younger age (20-30), which might have influenced our results. In contrast, we do not consider our decision to offer PM UIs only in English or German a limiting factor, since less than 3% of the participants with one of our target PMs used a different language in their browser (see Section 6.3).

8 Related Work

The security of password managers has been extensively studied, and several design and implementation vulnerabilities have been discovered in their web applications [30, 44] and their desktop counterparts [34]. Virtually every aspect of password managers received attention, from the format used to store passwords [20] to their password generation algorithms [34]. The interaction between browsers and password managers was especially scrutinized; several attacks demonstrated how to recover users' passwords by abusing auto-filling features [34, 42] or XSS attacks [44]. Even mobile versions of password managers were attacked with similar vectors [7]. Our attack is different as it does not exploit password managers themselves; rather, it leverages ambiguity in browser UIs to confuse users.

Despite password managers being recommended to increase users' online security, their adoption has been slow due to usability issues [15], differences in reason for adoption (e.g., convenience versus security) [36], and mistrust in web-based features such as synchronization [25, 37] and password auditing to detect compromised and insecure passwords [35]. Older password managers suffered from user misunderstandings and wrong mental models, even worsening users' security behavior [15].

While password managers effectively help people improve their passwords [47], browser-based password managers have been observed to encourage password reuse and the use of weak passwords [32, 36], especially due to lacking password generation features for a long time. Since introducing such features, browser-based password managers now also successfully nudge users towards the use of secure random passwords [48]. However, despite the support of password managers, random passwords are still difficult to manage for users who are concerned about remembering and entering them when the manager is unavailable [35].

Finally, the reliance of password managers on a master password used for authentication, decryption of credentials, and especially login to their web-based versions is a known drawback [22, 33], with solutions leveraging multiple devices instead [33], separation of concerns between master key and login password [22], or storing only less sensitive credential helpers [43].

9 Conclusion

In this paper, we show that phishing attacks targeting password manager browser extensions are a real threat: more than 30% of password manager users in a real-world large-scale study with close to 30,000 participants fell for our impersonations and revealed their master password.

This attack differs from past phishing attacks against password managers in several ways: (i) it highlights and exploit the users' confusion regarding which parts of a browser UI are trusted and which ones are not; (ii) master passwords of password manager are valuable secrets also used as login passwords to the online vault by many managers; and (iii) it shifts user focus away from the password manager to a decoy website the victim trusts or perceives as low-risk and, by exploiting this trust, compromises their password manager.

The last element makes this attack particularly threatening, as it could be launched from *legitimate* but compromised websites, depriving the victim of all indicators to detect the attack. This is a serious implication that forces users to be extra careful, since any website could imitate their password manager anytime.

Our analysis of potential countermeasures highlights further challenges: first, despite attempts to improve the distinction between trusted and untrusted UI elements of web browsers [40], extensions still do not respect these principles and let trusted UI elements overlap the viewport. However, it is unclear if even drastic measures, such as moving authentication workflows completely into the trusted browser chrome, would prevent users from falling for similar forgeries in the viewport. Further, our results validated that personalized UI elements, such as profile pictures, are unlikely to be effective, as their absence does not alert users to the deception. Finally, multi-factor authentication may raise the bar for attackers but can be defeated with *runtime phishing* [45]. Even phishing-resistant factors, such as passkeys, are affected by downgrade attacks, as they still rely on the master password as a fall-back or recovery mechanism.

10 Acknowledgements

This research was supported by the Zurich Information Security Center (ZISC).

Open Science

We recorded password metadata and mouse movements of all website visitors. While we assured our participants that all data would be anonymized, we did not mention the possibility of publication in our debriefing material. Publishing individual records without consent would not be compatible with our institution's ethical standards, as it could have affected participants' opt-out decision.

Ethics Considerations

Our study was approved by our institution's IRB and executive board. Due to the sensitive nature of our experiment, the project was reviewed by and/or conducted in collaboration with the university's Legal Office, Human Resources, Student Administration, IT, and Corporate Communications. In the following, we will summarize the key points of our IRB application.

User deception and waiver of informed consent. Our study is a "natural observational" study, the preferred way of setting up ecologically valid experiments about phishing detection and perception [19, 26]. Both real phishing attacks and simulated phishing campaigns rely on user deception to be successful; any forewarning will likely lead to additional diligence on the participant's part. The effect would be skewed results that do not represent the considered population's reaction in the face of a real attack. Consequently, it is common practice not to inform participants of a phishing simulation beforehand [19, 26, 38]; Our IRB granted us a waiver of informed consent after assessing that the study respected the necessary criteria (study could not have been conducted otherwise; minimal risk for participants welfare, and full debriefing after the study ended). Furthermore, participants were informed during the debriefing of their right to opt-out, which would lead to deleting all their data.

Recruitment of Participants. Following our IRB application, we encouraged the corresponding university offices to exclude vulnerable people from the study. We only involved participants whose contact data was provided to us by our university.

Privacy and Data Protection. Every participant was represented with a pseudonym based on an HMAC with a secret key. However, we conducted the evaluation anonymously. To protect against data breaches, we kept demographic information separate from the study infrastructure and stored all recorded data under pseudonyms. After submission, all data will be irrevocably anonymized by deleting the secret key.

Data Collection and Passwords. We did not collect any passwords, but discarded all user inputs directly in the participant's web browser. We only saved password metadata (like length and similarity between consecutive login attempts) in our system, and took a heuristic decision about the password's correctness. While we stored client information like browser language or User-Agent, we treated this information confidentially, and it is only associated with the aforementioned pseudonym. Furthermore, we informed participants in the debriefing in detail about what we collected and reassured them that all their data was safe.

Risks and Countermeasures. We identified the following risks in our IRB application: (i) data breach, (ii) emotional distress of participants, (iii) reputational damage for the institution, and (iv) a higher workload for IT services due to the phishing simulation. We addressed all these risks, namely (i) by employing encryption, pseudonymization, and strict system monitoring during the campaign, (ii-iv) by designing a thorough and timely debriefing of all participants, and (iii/iv) by involving corporate communications before and during the simulation. After the study, we gave an interview explaining the study in detail.

Phishing Mail. Our phishing simulation was different in the sense that our main interest was not how participants reacted to the email, but how they behaved on the phishing website. Consequently, we used a high-quality phishing email about public transport reimbursements after careful consideration and approval by our IRB.

Debriefing. Due to the scale of our study, face-to-face debriefings were impractical. Instead, we debriefed our participants with an email and a website. To maximize credibility, we sent the email from an email address associated with our IT services and hosted the website on the official web portal of our institution. The contents of both email and website were reviewed by our IRB, corporate communications, and representatives of IT services, and improved over several iterations.

Email. The purpose of the email was to provide all participants with a self-contained yet concise debriefing (around 460 words), regardless of their participation or performance. The email can be found in Section A.2.

Website. The debriefing website led with the same content as the email, followed by an additional 1,500-word Q&A for participants with further questions or a special interest. The Q&A contained a total of 12 questions on the topics of phishing, password managers, and our project. The questions can be found at the end of the email in Section A.2.

Debriefing Process. We ensured that the debriefing email was delivered to all participants. The debriefing process is detailed in Section 5.3.

Reception. Nine participants opted out of our study. In addition, 6 participants sent us complaints by email, and around 10 complained to the authors over other channels. On the other hand, we received around 10 emails or calls of participants showing appreciation or giving positive feedback, with another 23 asking questions about phishing or our study.

References

[1] Autofill From Browser Extensions | Bitwarden — bitwarden.com. https://bitwarden.com/help/autofill-browser/#inline-autofill-menu. [Accessed 02-06-2025].

[2] 1Password. Passkeys in 1password. https://support.1password.com/passkeys/, 2023. Accessed: 2025-01-21.

[3] 1Password. Two-factor authentication with 1password. https://support.1password.com/two-factor-authentication/, 2023. Accessed: 2025-01-21.

[4] 1Password. About your secret key. https://support.1password.com/secret-key-security/, 2025. Accessed: 2025-05-13.

[5] Sara Albakry, Kami Vaniea, and Maria K Wolters. What is this url's destination? empirical evaluation of users' url reading. In *Proceedings of the 2020 CHI conference on human factors in computing systems*, pages 1–12, 2020.

[6] Kholoud Althobaiti, Ghaidaa Rummani, and Kami Vaniea. A review of human-and computer-facing url phishing features. In *2019 IEEE European symposium on security and privacy workshops (EuroS&PW)*, pages 182–191. IEEE, 2019.

[7] Simone Aonzo, Alessio Merlo, Giulio Tavella, and Yanick Fratantonio. Phishing attacks on modern android. In *Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security*, pages 1788–1801, 2018.

[8] Bitwarden. Bitwarden release notes - version 2023.3.0. https://bitwarden.com/help/releasenotes/#2023.3.0, 2023. Accessed: 2025-01-21.

[9] Bitwarden. How to enable two-step login. https://bitwarden.com/learning/enable-two-step-login/, 2023. Accessed: 2025-01-21.

[10] Bitwarden. How to login with passkeys. https://bitwarden.com/help/login-with-passkeys/, 2023. Accessed: 2025-01-21.

[11] BleepingComputer. Bitwarden password vaults targeted in google ads phishing attack. https://www.bleepingcomputer.com/news/security/bitwarden-password-vaults-targeted-in-google-ads-phishing-attack/, 2023. Accessed: 2025-01-21.

[12] Cristian Bravo-Lillo, Lorrie Cranor, Julie Downs, Saranga Komanduri, Stuart Schechter, and Manya Sleeper. Operating system framed in case of mistaken identity: measuring the success of web-based spoofing attacks on os password-entry dialogs. In *Proceedings of the 2012 ACM conference on Computer and communications security*, pages 365–377, 2012.

[13] Sean Cassidy. Lostpass. https://www.seancassidy.me/lostpass.html, 2016.

[14] Sen Chen, Lingling Fan, Chunyang Chen, Minhui Xue, Yang Liu, and Lihua Xu. Gui-squatting attack: Automated generation of android phishing apps. *IEEE Transactions on Dependable and Secure Computing*, 18(6):2551–2568, 2019.

[15] Sonia Chiasson, Paul C van Oorschot, and Robert Biddle. A usability study and critique of two password managers. In *USENIX Security Symposium*, volume 15, pages 1–16, 2006.

[16] Dashlane. 2-factor authentication (2fa) in dashlane. https://support.dashlane.com/hc/en-us/articles/202625042-2-factor-authentication-2FA-in-Dashlane, 2023. Accessed: 2025-01-21.

[17] Periwinkle Doerfler, Kurt Thomas, Maija Marincenko, Juri Ranieri, Yu Jiang, Angelika Moscicki, and Damon McCoy. Evaluating login challenges as adefense against account takeover. In *The World Wide Web Conference*, WWW '19, page 372–382, New York, NY, USA, 2019. Association for Computing Machinery.

[18] Farida Eleshin, Qi Sun, Mengzhe Ye, Sauvik Das, and Jason I Hong. Of secrets and seedphrases: Conceptual misunderstandings and security challenges for seed phrase management among cryptocurrency users. In *Proceedings of the 2025 CHI Conference on Human Factors in Computing Systems*, pages 1–19, 2025.

[19] Peter Finn and Markus Jakobsson. Designing ethical phishing experiments. *IEEE Technol. Soc. Mag.*, 26(1):46–58, 2007.

[20] Paolo Gasti and Kasper B Rasmussen. On the security of password manager database formats. In *Computer Security–ESORICS 2012: 17th European Symposium on Research in Computer Security, Pisa, Italy, September 10-12, 2012. Proceedings 17*, pages 770–787. Springer, 2012.

[21] Tom N Jagatic, Nathaniel A Johnson, Markus Jakobsson, and Filippo Menczer. Social phishing. *Communications of the ACM*, 50(10):94–100, 2007.

[22] Hyeonhak Jeong and Hyunggu Jung. Monopass: a password manager without master password authentication. In *Companion Proceedings of the 26th International Conference on Intelligent User Interfaces*, pages 52–54, 2021.

[23] Mohammed Jubur, Prakash Shrestha, and Nitesh Saxena. An in-depth analysis of password managers and two-factor authentication tools. *ACM Computing Surveys*, 57(5):1–32, 2025.

[24] Nikolaos Karapanos, Claudio Marforio, Claudio Soriente, and Srdjan Capkun. {Sound-Proof}: Usable {Two-Factor} authentication based on ambient sound. In *24th USENIX security symposium (USENIX security 15)*, pages 483–498, 2015.

[25] Ambarish Karole, Nitesh Saxena, and Nicolas Christin. A comparative usability evaluation of traditional password managers. In *Information Security and Cryptology-ICISC 2010: 13th International Conference, Seoul, Korea, December 1-3, 2010, Revised Selected Papers 13*, pages 233–251. Springer, 2011.

[26] Daniele Lain, Kari Kostiainen, and Srdjan Capkun. Phishing in organizations: Findings from a large-scale and long-term study. In *43rd IEEE Symposium on Security and Privacy, SP 2022, San Francisco, CA, USA, May 22-26, 2022*, pages 842–859. IEEE, 2022.

[27] Leona Lassak, Elleen Pan, Blase Ur, and Maximilian Golla. Why aren't we using passkeys? obstacles companies face deploying FIDO2 passwordless authentication. In Davide Balzarotti and Wenyuan Xu, editors, *33rd USENIX Security Symposium, USENIX Security 2024, Philadelphia, PA, USA, August 14-16, 2024*. USENIX Association, 2024.

[28] LastPass. Passwordless authentication features. https://www.lastpass.com/features/passwordless-authentication, 2023. Accessed: 2025-01-21.

[29] LastPass. Two-factor authentication. https://www.lastpass.com/solutions/authentication/two-factor-authentication, 2023. Accessed: 2025-01-21.

[30] Zhiwei Li, Warren He, Devdatta Akhawe, and Dawn Song. The {Emperor's} new password manager: Security analysis of web-based password managers. In *23rd USENIX Security Symposium (USENIX Security 14)*, pages 465–479, 2014.

[31] Luka Malisa, Kari Kostiainen, and Srdjan Capkun. Detecting mobile application spoofing attacks by leveraging user visual similarity perception. In *Proceedings of the Seventh ACM on Conference on Data and Application Security and Privacy*, pages 289–300, 2017.

[32] Peter Mayer, Collins W Munyendo, Michelle L Mazurek, and Adam J Aviv. Why users (don't) use password managers at a large educational institution. In *31st USENIX Security Symposium (USENIX Security 22)*, pages 1849–1866, 2022.

[33] Daniel McCarney, David Barrera, Jeremy Clark, Sonia Chiasson, and Paul C Van Oorschot. Tapas: design, implementation, and usability evaluation of a password manager. In *Proceedings of the 28th annual computer security applications conference*, pages 89–98, 2012.

[34] Sean Oesch and Scott Ruoti. That was then, this is now: A security evaluation of password generation, storage, and autofill in browser-based password managers. In *Proceedings of the 29th USENIX Conference on Security Symposium*, pages 2165–2182, 2020.

[35] Sean Oesch, Scott Ruoti, James Simmons, and Anuj Gautam. "it basically started using me:" an observational study of password manager usage. In *Proceedings of the 2022 CHI Conference on Human Factors in Computing Systems*, pages 1–23, 2022.

[36] Sarah Pearman, Shikun Aerin Zhang, Lujo Bauer, Nicolas Christin, and Lorrie Faith Cranor. Why people (don't) use password managers effectively. In *Fifteenth Symposium on Usable Privacy and Security (SOUPS 2019)*, pages 319–338, 2019.

[37] Hirak Ray, Flynn Wolf, Ravi Kuber, and Adam J Aviv. Why older adults (don't) use password managers. In *30th USENIX Security Symposium (USENIX Security 21)*, pages 73–90, 2021.

[38] David B. Resnik and Peter R. Finn. Ethics and phishing experiments. *Sci. Eng. Ethics*, 24(4):1241–1252, 2018.

[39] Joshua Reynolds, Deepak Kumar, Zane Ma, Rohan Subramanian, Meishan Wu, Martin Shelton, Joshua Mason, Emily Stark, and Michael Bailey. Measuring identity confusion with uniform resource locators. In *Proceedings of the 2020 CHI Conference on Human Factors in Computing Systems*, pages 1–12, 2020.

[40] Thomas Roessler and Anil Saldhana. Web security context: User interface guidelines. W3C recommendation, W3C, August 2010. https://www.w3.org/TR/2010/REC-wsc-ui-20100812/.

[41] Stuart E Schechter, Rachna Dhamija, Andy Ozment, and Ian Fischer. The emperor's new security indicators. In *2007 IEEE Symposium on Security and Privacy (SP'07)*, pages 51–65. IEEE, 2007.

[42] David Silver, Suman Jana, Dan Boneh, Eric Chen, and Collin Jackson. Password managers: Attacks and defenses. In *23rd USENIX Security Symposium (USENIX Security 14)*, pages 449–464, 2014.

[43] Elizabeth Stobert and Robert Biddle. A password manager that doesn't remember passwords. In *Proceedings of the 2014 New Security Paradigms Workshop*, pages 39–52, 2014.

[44] Ben Stock and Martin Johns. Protecting users against xss-based password manager abuse. In *Proceedings of the 9th ACM symposium on Information, computer and communications security*, pages 183–194, 2014.

[45] Enis Ulqinaku, Hala Assal, AbdelRahman Abdou, Sonia Chiasson, and Srdjan Capkun. Is real-time phishing eliminated with {FIDO}? social engineering downgrade attacks against {FIDO} protocols. In *30th USENIX Security Symposium (USENIX Security 21)*, pages 3811–3828, 2021.

[46] Enis Ulqinaku, Daniele Lain, and Srdjan Capkun. 2fapp: 2nd factor phishing prevention. In *Proceedings of the 12th Conference on Security and Privacy in Wireless and Mobile Networks, WiSec 2019, Miami, Florida, USA, May 15-17, 2019*, pages 60–70. ACM, 2019.

[47] Samira Zibaei, Dinah Rinoa Malapaya, Benjamin Mercier, Amirali Salehi-Abari, and Julie Thorpe. Do password managers nudge secure (random) passwords? In *Eighteenth Symposium on Usable Privacy and Security (SOUPS 2022)*, pages 581–597, 2022.

[48] Samira Zibaei, Amirali Salehi-Abari, and Julie Thorpe. Dissecting nudges in password managers: simple defaults are powerful. In *Nineteenth Symposium on Usable Privacy and Security (SOUPS 2023)*, pages 211–225, 2023.

A Additional Study Information

A.1 Phishing email

In our phishing simulation, we advertised public transport reimbursements. We picked this topic as a good compromise between generating sufficient interest while still being ethically justifiable. We worked in cooperation with the institution and its IRB to design the phishing email, which was sophisticated and akin to official communication. For the hostnames, we leveraged a subtle typosquat URLs resembling the institution's official domain.

A.1.1 Content (English version only)

To: Students and members of <*redacted*>

Dear <*redacted*>,

The cost of public transport has been rising over the past years. For example, the price of a <*redacted*> increased from <*redacted*> in 2020 to <*redacted*> in 2024, and the price for a <*redacted*> reached <*redacted*> this year.
To make public transport more affordable, we are excited to announce <*redacted*>, a project aimed at supporting the <*redacted*> community by subsidizing public transport.

Key information:

- <*redacted*> members and students may request partial reimbursements.
- Reimbursements apply to ticket purchases in the year 2024.
- Eligibility requirements and the reimbursed amounts are detailed on the project page.
- While there is no fixed deadline, please be aware that this is a pilot project and funds are currently limited by a donation. You can check your eligibility and submit your application with a few clicks on the following website (requires <*redacted*> login):

https://<phishing link>

Kind regards,
<*redacted*>
<*redacted*>
Transport and Traffic

A.2 Debriefing email

Dear <*redacted*> member,

In the past few days, you may have received an email with the subject line "Announcement: Public transport reimbursements". It introduced "<*redacted*>" and offered reimbursement options.

This was a phishing simulation, a simulated cyber attack. Regrettably, we must inform you that no reimbursements are available: the mentioned email, allegedly from "<*redacted*>", was in fact sent by "info@ <*redacted*>". This email and the <*redacted*> website it linked to were imitations, showing how cybercriminals could try to steal your credentials in a phishing attack.

All your passwords are safe. Your security is our highest priority. Rest assured that nobody has learned your password(s). We discarded any data you may have entered on the phishing website (or into the faked password manager, see below), and none of your data left your device.

This phishing simulation was reviewed by the <*redacted*> Ethics Commission and the Data Protection Officer, and it was authorized by the <*redacted*> Executive Board. It is

being carried out by <*redacted*> researchers in collaboration with <*redacted*> (IT Services).

You may have been asked to unlock your password manager. In addition to the <*redacted*> login form, the website may have imitated your password manager (if you use one) and asked for your master password. This is part of a study conducted by <*redacted*>.

Your participation is confidential. We do not know if you entered your real password or not, as we have not collected sensitive data. Furthermore, we will evaluate the results of this simulation anonymously. <*redacted*> will not learn about the performance of any individual participants, and your behavior in the phishing simulation will not have any consequences for you.

We imagine you may still have questions... which is why we compiled a Q&A below. It explains in more detail what phishing is, how you can protect yourself against phishing emails, and why we conducted this simulation. If it leaves any questions unanswered, feel free to contact us via <*redacted*>.

... and we also have a few questions for you. We, the researchers behind this project, would be incredibly grateful if you could take a few minutes to complete a short survey. Your responses will be evaluated anonymously and will contribute to our study's findings. The survey is hosted on the <*redacted*> platform and be accessed via the following link.

To the survey (link)
Thank you!

Please do not inform your peers just yet. This simulation allows participants to test their phishing awareness in a secure environment. The simulation ends on October 31, and we would appreciate your discretion until that date.

We thank you for your understanding and your contribution to our research!

Kind regards,
<*redacted*>

If you would like to learn more about this project, we recommend our debriefing Q&A at <*redacted*>. Alternatively, you may use any of the links below to jump directly to the answers of specific questions.

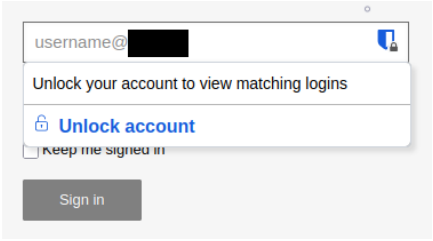
About Phishing
- What is Phishing?
- What is the purpose of a phishing simulation?
- How can I detect Phishing?

About Password Managers
- What is a password manager?
- Does a password manager protect against phishing?
- Should I use a password manager or not?

About this Project
- What is the goal of this project?
- Who is responsible?
- Why did you not ask me beforehand if I wanted to participate?
- What data have you collected about me and how is it protected?
- I want you to delete my data.

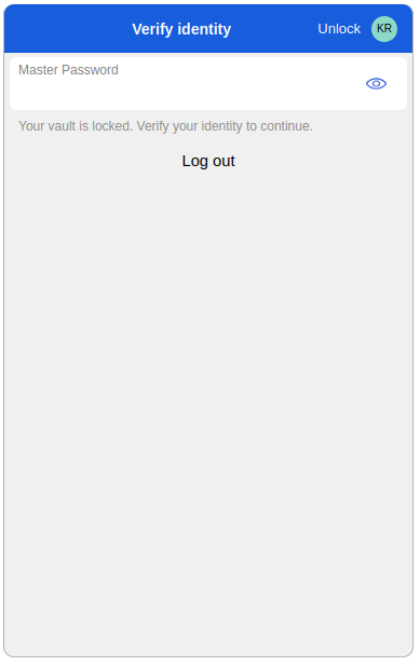
A.3 Survey

- Q1: Do you know your <*redacted*> password by heart? (yes/no)
- Q2: Please select the password manager you use the most for storing website passwords. (single choice + free text)
- Q3: Do you store your <*redacted*> password in this password manager? (yes/no)
- Q4: How often do you use the browser extension of your password manager? (single choice)
- Q5: How often do you interact with the desktop application of this password manager? Interactions include unlocking your vault, using or changing passwords, etc. (single choice)
- Q6: How often do you login on the website of your password manager to access your passwords (i.e., use the online vault)? (single choice)
- Q7: How do you usually unlock your browser extension? By providing my credentials ...' (multiple choice)
- Q8: If you visit a website and your password manager is locked. What is your first step? I start by ...': (multiple choice)
- Q9: How often do you have to unlock your password manager? (single choice)
- Q10: What is the primary credential you use to unlock your password manager? (single choice)
- Q11: Which of the following steps of the phishing simulation did you complete? Your selection should indicate the last completed step (e.g., if you read the email but did not follow the link, choose 'I read the email.'). (single choice)
- Q12: On the website, did you use or attempt to use your password manager to enter your <*redacted*> password?' (yes /no)
- Q13: Regarding your password manager, did you notice anything out of the ordinary on the website?' (open)
- Q14: Do you have any other remarks or feedback? (open)

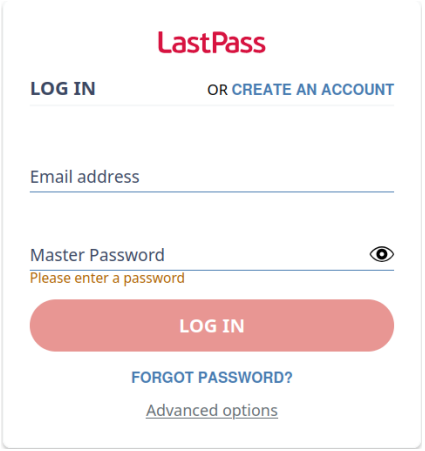


(a) Bitwarden input decorator

(b) LastPass input decorator

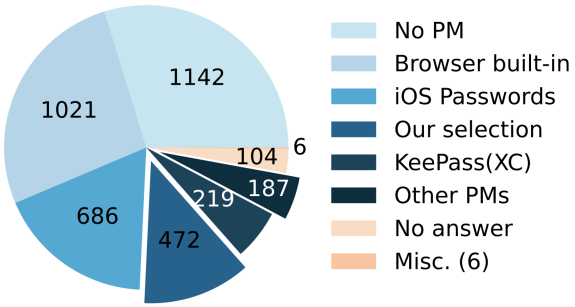


(c) Bitwarden login of group $\mathbf{B}_{Ext}$.

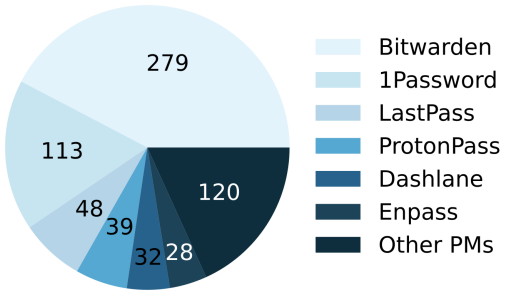


(d) LastPass login of group $\mathbf{L}_{Ext}$.

Figure 7: **Examples of UIs used in our study.**



(a) High-level password manager distribution. The group "our selection" equals to the sum of the password managers we were interested in. Together with "other PMs" it is broken down in Figure 8b.



(b) Detailed distribution of exploded slices of Figure 8a. All products aggregated in "Other" have been reported less than 20 times.

Figure 8: **Usage of password managers.** Responses to the question *Please select the password manager you use the most for storing website passwords.*

Table 3: **PM phishing success rate**, per age group.

	≤ 30	31 - 40	41 - 50	> 50	Total
Bitwarden	37/157 (23.57%)	10/33 (30.30%)	3/16 (18.75%)	5/13 (38.46%)	55/219 (25.11%)
LastPass	2/36 (5.56%)	1/15 (6.67%)	0/5 (0.0%)	0/1 (0.00%)	3/57 (5.26%)
1Password	38/84 (45.24%)	14/29 (48.28%)	6/13 (46.15%)	4/12 (33.33%)	62/138 (44.93%)
Dashlane	15/25 (60.0%)	4/7 (57.14%)	1/1 (100.0%)	0/1 (0.00%)	20/34 (58.82%)
Total	92/302 (30.46%)	29/84 (34.52%)	10/35 (28.57%)	9/27 (33.33%)	140/448 (31.25%)