Leaked in Translation: Measuring and Mitigating Sensitive Data Exposure to Third-Party Services via Browser-Based Translation

Tianlang Xie
tianlang.xie@ru.nl
Radboud University
Nijmegen, The Netherlands

Xavier de Carné de Carnavalet
xavier.carnavalet@ru.nl
Radboud University
Nijmegen, The Netherlands

Abstract

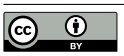
Webpage translation is a built-in feature of modern browsers and offered via translation extensions. When a user triggers full-page translation, the tool decides which content to extract and send for processing. Users may reasonably expect invariant strings such as email addresses and phone numbers to remain local, yet little is known about what translation tools actually transmit to cloud-based services. We present a measurement of network requests issued by four major browsers and a popular translation extension when performing full-page translation on authenticated pages from 23 Dutch websites. Across 28 real-world pages showcasing real or dummy sensitive information, we find that translation tools make little effort to exclude sensitive data that require no translation, except for values held in standard HTML forms. The translated domain, and sometimes the full URL, is often exposed to the translation provider. We then study the use of translation opt-out markers and find that they are rarely used to protect sensitive values. Furthermore, the standard `translate="no"` attribute is respected by the browsers but not by the tested extension. Finally, we build and evaluate a browser extension that proactively marks likely sensitive values before translation. On Chrome, our extension reduced the inclusion rate of sensitive data by 48.2%; an optional masking mode removes sensitive data before translation with tools that do not respect the opt-out attribute. Our results highlight the privacy risks of free automated translation features embedded in or added to browsers. It remains unclear who should prevent these exposures: users typically cannot see what the tool sends, websites have no agreed duty to stop a user's own translation tool from transmitting sensitive content, and opt-out markers are not designed as privacy controls. On-device translation, as in Firefox, avoids the issue by keeping page content local.

CCS Concepts

• **Security and privacy** → **Privacy protections**; *Browser security*; Human and societal aspects of security and privacy; • **Information systems** → Web applications.

Keywords

Webpage translation, web privacy measurement, sensitive data exposure, browser extensions, machine translation, opt-out markers



This work is licensed under a Creative Commons Attribution 4.0 International License.
WPES '26, The Hague, Netherlands
© 2026 Copyright held by the owner/author(s).
ACM ISBN 979-8-4007-3026-9/2026/11
https://doi.org/10.1145/3847192.3847371

ACM Reference Format:
Tianlang Xie and Xavier de Carné de Carnavalet. 2026. Leaked in Translation: Measuring and Mitigating Sensitive Data Exposure to Third-Party Services via Browser-Based Translation. In *25th Workshop on Privacy in the Electronic Society (WPES '26), November 15–19, 2026, The Hague, Netherlands.* ACM, New York, NY, USA, 13 pages. https://doi.org/10.1145/3847192.3847371

1 Introduction

Automated webpage translation plays an important role in enabling users to access information written in foreign languages [15]. It allows people to read news articles, technical documentation, e-commerce sites, and public service pages that would otherwise remain inaccessible, particularly when traveling or living abroad. Such capabilities are integrated into modern browsers, which can automatically detect the language of a page and prompt the user to translate it [11]. When the user accepts, the browser typically extracts the page's textual content and processes it either locally or sends it to a remote translation service, which returns a translated version that is rendered in place.

Free automated translation services raise privacy concerns [15]. The text users submit to these services can contain personal data, either relating to the user or to other parties, and may even include sensitive information (e.g., health-related details). Providers then typically retain the submitted content and reserve broad rights to reuse it, for example to improve their models or for commercial purposes such as advertising.

While the privacy risks of free online machine translation services have been discussed from a legal perspective, there is limited empirical work measuring sensitive data exposure caused by browser-based full-page translation features. In the browser setting, the risks are indeed amplified. Unlike a dedicated translation website where users consciously paste selected text, the browser feature introduces ambiguity about exactly which content is transmitted. Users may reasonably assume that only the text requiring translation is sent, while invariant strings such as email addresses, phone numbers, or other personal identifiers are left untouched. In practice, however, a straightforward/privacy-invasive extraction method may simply transmit the entire page content. Moreover, users have no practical way to edit or sanitize the page before translation. While it is possible to manually select only non-sensitive text and translate it piece by piece, this approach negates the usability advantages of the one-click full-page translation feature.

This concern is particularly relevant for personalized page content, such as what appears on authenticated parts of a website. A user profile page typically includes names, email addresses, and other personal details. Healthcare websites, e.g., accessed when visiting healthcare providers in another country, could include explicit

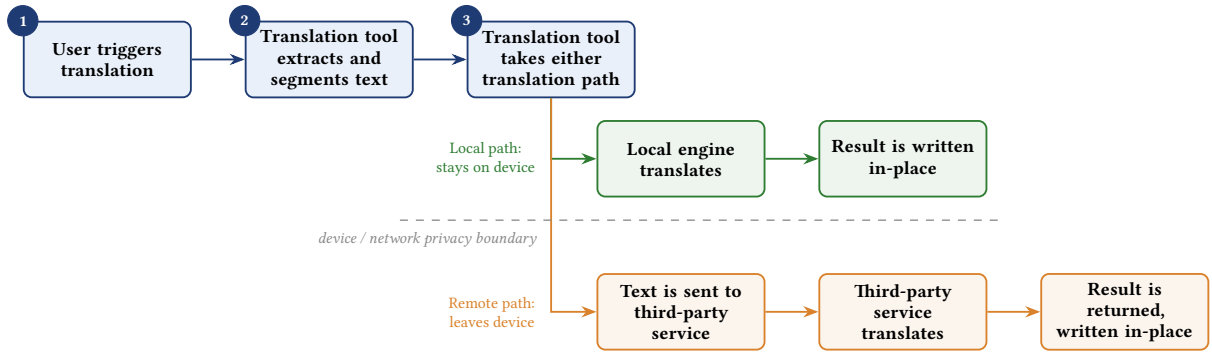


Figure 1: Simplified pipeline of browser-based webpage translation

appointment reminders and laboratory results tied to the user's full name and identifiers.

In this paper, we study how browser-based translation tools manipulate sensitive page elements and which ones leak to remote servers when leveraging their full-page translation feature. We consider the translate button of the top five major desktop browsers (Firefox, Chrome, Brave, Edge, Opera) on Windows and one popular Chrome-based extension, Immersive Translate, with 3M users. We follow a black-box approach where we do not assume access to the browser, extension, or backend source code, and instead observe the rendered page before translation, the network requests occurring during translation, and whether sensitive values appear in the captured data. We selected a corpus of Dutch websites with an authenticated section we can log in to that displays account details or other sensitive data, using the authors' existing accounts and/or dummy accounts with synthetic values we create. These websites span various categories, including e-commerce, public transportation, government, education, and financial/payment websites. We translate the pages from Dutch to English.

Contributions. This paper makes the following contributions.

(1) We construct a corpus of 23 Dutch websites and 28 authenticated pages that display account, service, or transaction data, and measure which sensitive values are sent to remote servers during one-click full-page translation. Across 135 comparable runs, the overall inclusion rate is 59.2%. The HTML tag encapsulating such values made noticeable differences between translation tools, notably with standard form controls, but the text itself did not influence processing (Section 5).
(2) We analyze the structure of translation requests and the metadata they carry. Built-in tools unnecessarily expose the source domain and, depending on the website's referrer policy, sometimes the full page URL, leaking browsing context that is not required for translation. Although text is split across multiple API calls, this fragmentation did not appear to be intended for isolating sensitive values and therefore did not reduce the privacy exposure (Section 6).
(3) We investigate the prevalence and usefulness of opt-out markers, i.e., HTML attributes aimed at preventing the translation of selected elements, on our website corpus. They are rarely used, and we find that only the standard `translate="no"` attribute is effective, and only for tools that implement this feature: the five built-in tools respect it, while the Immersive Translate extension does not (Section 7).
(4) We design and evaluate a browser extension, Local Translate Guard, that gives users a practical way to sanitize a page before translation by automatically marking likely sensitive values with `translate="no"`. The goal is to reduce unnecessary privacy exposures when users cannot reasonably edit the page by hand. On Chrome, the extension lowered the inclusion rate by 48.2% (from 63.3% to 32.8%). This is not a complete defense: detection is imperfect and some tools ignore the marker. We also propose an optional masking mode that removes sensitive data before translation when the opt-out attribute is not respected (Section 8). We make the tool available at https://github.com/black5ugar/translate_guard.
(5) We offer immediate recommendations for users, browser/extension vendors, but also discuss the responsibly of each party involved, as there do not appear to be a straightforward answer to the privacy problem of automated translation (Section 9), except resorting to on-device translation.

2 Background and threat model

How browser-based translation works. Modern browsers can help users translate whole webpages. After automatically detecting the source language, they can prompt users to translate a foreign document into the user's preferred language. Upon confirmation, the browser extracts translatable text, segments it, sends it to a local translation engine or a remote server, and writes the result back into the DOM [22, 34]. We illustrate this process in Figure 1.

To avoid unnecessary processing, only text that is visible to the user is translated at once. After the user scrolls down a page or opens a hidden menu, later content is dynamically translated on the go. What constitutes visible text is tool-dependent.

Historically, browsers have relied on cloud services for translation, in which the extracted page text leaves the device, possibly resulting in privacy leakages to third parties. Starting from v118, Firefox relies on a local translation engine [24, 25].

When full-page translation has been activated for a page, some browsers (e.g., Chrome, Brave, Edge) continue to automatically translate subsequent pages upon navigation, even when leaving

Table 1: Documented backend services and retention policies for webpage translation tools

Tool	Backend / service	Remote processing	Data retained	Stated retention	Later use
Firefox	Firefox Translations	No	—	—	—
Chrome	Google Translate	**Yes**	n.s.	n.s.	n.s.
Brave	Brave Translate	**Yes**	None	Deleted after translation	No
Edge	Microsoft Translator	**Yes**	**Webpage text**	n.s.	Service improvement
Opera	Opera Translate	**Yes**	None	Not stored	No
Safari	Apple translation service	**Yes**	**URL metadata**[*]	**Up to 5 years**[*]	Service improvement[*]
Immersive Translate	User-selected engine	**Yes**[†]	Usage statistics and error logs[†]	n.s.[†]	Service improvement[†]

Notes. Entries summarize public documentation, not our network measurements. "n.s." means no specific statement; "—" means not applicable. [*]Safari entries concern URL metadata outside Private Browsing, not translated content. [†]Immersive Translate entries depend on the selected engine. Sources: Chrome [12, 34]; Edge [18, 20]; Opera [27]; Brave [4]; Firefox [26]; Safari [2]; Immersive Translate [13, 14].

the original website. This behavior seems convenient; however, it can introduce unwanted leakages when a new page unexpectedly displays sensitive information the user wishes not to be sent to a third party. By the time the user notices, it is too late.

Browsers also offer to "Always translate pages in" a certain language [12], triggering translation whenever a page language matches. These two automatic mechanisms reinforce the uncertainty the user has over what content is eventually processed and submitted to an online translation service.

Separate browser extensions can offer a translation service as well. We confirmed Immersive Translate for Chrome does support the two automation features discussed above. Others may require a paid subscription (e.g., DeepL); we did not evaluate such extensions.

Translation opt-out markers. The W3C provides a standard way to mark content that should not be translated: the HTML5 global attribute `translate` [17, 38]. Elements marked `translate="no"` should not be modified by translation or localization, and the value is inherited by child elements unless overridden. This gives developers a standard mechanism to define translation boundaries within a page. Both Google and Microsoft have supported `class="notranslate"`, but this behavior is considered legacy [38]. Google also offers its own meta tag (`name="google" content="notranslate"`) that prevents Chrome from offering a translation. Note that opt-out attributes on specific elements is designed to prevent unwanted changes to the page, e.g., the W3C suggests that software code could be marked as non-translatable, while code comments could be translatable, or text in one language with quotes in other one would benefit from the `translate` attribute [38]. These markers are, therefore, not designed to protect possible privacy leakages. Nonetheless, we examine whether websites use these markers for sensitive data and whether translation tools respect them.

Threat model and scope. We study full-page translation features offered by desktop browsers as well as browser extensions (hereafter referred to as *translation tools*). We consider text to be sensitive if it is personally identifiable information (PII), such as names, email addresses, physical addresses, location data, national or social identifiers, and birth dates; or personal details that are either numerical (e.g., financial data) or time-based (e.g., location history). Importantly, such information typically does not require translation and thus, its inclusion in a remote translation request is unnecessary.

We define a privacy leakage as any instance in which such sensitive information or other sensitive metadata about the user leaves the device and is transmitted to a remote server for translation.

We do not consider network adversaries. We assume that the browser/extension vendor and the remote third-party translation provider are either 1) honest, and follow their stated policy, which could be abusive with respect to a user's privacy; or 2) honest-but-curious: they follow their stated policies but may still exploit the received sensitive information for their own purposes, potentially contrary to users' reasonable privacy expectations when using the translation feature.

Documented backend handling. The impact of sensitive information leakage is modulated by the translation service provider's privacy policy. We reviewed public privacy policies and product documentation for each tool and summarized them in Table 1. Firefox is the clearest local case: Since 2023, Mozilla states that translation happens on-device and page text is not sent to cloud servers [24]. Brave promises to delete text after translation, while Opera states that it does not store translation data [4, 27]. Safari discards translated text but may store the page URL for up to five years outside Private Browsing [2]. Chrome and Edge give less specific statements about text retention [12, 20]. Edge states that Azure Speech in Foundry Tools is used to remove identifiers (server-side) before the text is stored for service improvement [20]. Immersive Translate states that webpage text is sent to selected translation services [14], preserves data for as long as necessary. No privacy policies stated they would share the data with third parties for other reasons than providing translation services, or for advertising/analytics.

3 Related work

Privacy of machine translation. Prior work treats machine translation as a data-protection problem. Kamocki et al. [15] examine the privacy implications of free online machine translation services under EU data protection law. They analyze the data flows involved when users submit text for translation and highlight that service

providers often retain the submitted content and claim broad rights to reuse it for secondary purposes. Google's Terms of Service, for example, grant the company a worldwide license to use, reproduce, modify, and distribute the content for the purposes of operating and improving its services, as well as developing new ones [10]. The authors argue that users are frequently unaware of these practices and that the processing of personal data (including data relating to third parties) raises significant challenges regarding consent and lawfulness. Vieira et al. [37] presents a user study in which users view machine translation as beneficial for privacy, as human translators are kept out of the loop. However, users tended to think about privacy issues only after sensitive information was accidentally fed into the translation tool. Project Bergamot [23, 28] was explicitly motivated by these concerns and developed a local translation engine, now used in Firefox [25], to avoid sending page content to third-party servers.

Related web privacy works. Senol et al. [30] show that sensitive form data can reach third parties before submission due to third party scripts on the page. In our work, the focus shifts from the website developer's to the user's action, as user-initiated translation is independent of the website being visited. In a user survey by Utz et al. [36], third party scripts were often used by developers on their websites without proper privacy considerations. With browser-based translation tools, website developers have less control over what data is possibly exposed to third parties. Research on browser extensions shows that privileged access to the browsing context enlarges what an extension can read or send [16, 32, 40]. In our work, we do not assume translation extensions are malicious, but they may collect sensitive data as part of their operation.

Mitigation models. Translation-specific proposals such as PRISM protect on the user side [29], and local systems such as Bergamot and translateLocally reduce dependence on cloud processing [3, 9]. Data Loss Prevention systems offer an engineering model in which outbound actions are inspected or redacted by policy [1]. We contribute an empirical measurement of browser-based translation in realistic settings to characterize the extent of the problem, plus a client-side mitigation.

4 Methodology

4.1 Translation tool selection

We start from the top six desktop browsers reported by StatCounter as of February 2026: Chrome, Edge, Safari, Firefox, Opera, and Brave [33]. We rely on traffic capture from the browser Developer Tools' Network panel to inspect possible remote translation requests. We also consider the Immersive Translate extension on Chrome,[1] selected because it had the highest reported installation count among webpage translation extensions in the Chrome Web Store that provide free full-page translation at the time of selection. Given its (only) 3M users, compared with the hundred million or billion users of browsers, we do not investigate other less popular extensions. Immersive Translate's remote translation requests do not appear in the Developer Tools, so we capture them using mitmproxy [8].[2]

Table 2: Translation tools and their measurement method

Tool	Version	Method
Chrome built-in translation	147.0.7727.56	DevTools
Edge built-in translation	147.0.3912.60	DevTools
Opera built-in translation	130.0.5847.12	DevTools
Brave built-in translation	1.89.132	DevTools
Immersive Translate† (Chrome)	1.27.2	`mitmproxy`
Firefox Translations	149.0.2	DevTools
Safari webpage translation	26.4	*N/A*

Notes. †Using Immersive Translate's default backend (its own).

Safary. For Safari on macOS, we find that translation requests originate from the Apple-signed system daemon *translationd* that enforces certificate pinning, preventing our HTTPS interception attempts. The deamon is protected by the OS integrity and runtime security mechanisms; we did not try to defeat the OS protection to disable pinning and exclude Safari from our study.

We retrain the translators built in Firefox, Chrome, Brave, Edge, and Opera, and the Immersive Translate extension. Table 2 summarizes the tool versions and traffic capture methods used.

4.2 Website and webpage corpus

We focus on Dutch services so they can be translated to English, a realistic translation workflow for non-Dutch speakers in the Netherlands. We select services that (1) are based in or mainly serve the Netherlands, (2) are offered in Dutch only or the Dutch version in bilingual websites is translatable, and (3) offer an authenticated (login or government ID-protected) section where personal data is displayed. We filtered the Tranco list of 1 April 2026 [35] to 9,761 `.nl` domains and manually checked the 300 highest-ranked domains against our inclusion conditions. As a secondary source, we checked the top five services in each relevant Similarweb category [31] and added eligible Dutch services not already identified through Tranco. This screening produced 39 candidate services.

Accounts were often required for the tested pages. Many sites also bind accounts to a real identity or to a physical or purchase-based registration, as is typical of government services. Where the first author already had an account, we reused it; otherwise we created dummy accounts and populated them with synthetic values in realistic formats whenever registration was possible. Sites that we could not register for were excluded. The final corpus comprises 23 websites, grouped into five categories (see Table 3).

After login, we systematically explore the authenticated website to select one or multiple pages that include sensitive information, preferably the user profile page, and sometimes another page where relevant and sensitive information regarding the service is displayed (e.g., travel history). The corpus includes 28 test pages. We exclude the *PostNL* parcel tracking page in the main leakage measurements

[1] https://chromewebstore.google.com/detail/immersive-translate-ai-we/bpoadfkcbjbfhfodiogcnhhhpibjhbnh

[2] In retrospect, we could have inspected its service worker and observed the generated traffic directly within Chrome.

Table 3: Summary of the measured website corpus of authenticated (login/ID-protected) pages

Categories	Measured services	Tested pages	Typical sensitive information in measured pages
E-commerce / Marketplace	*Bol.com*; *Marktplaats*; *Amazon*; *Funda*; *Zalando*; *AH*; *Vinted*	9	account data; contact details; address data; orders; payment or price information
Transport / Travel	*Booking.com*; *TUI*; *NS*; *PostNL*; *Schiphol*; *TripAdvisor*; *Airbnb*; *OV-chipkaart*; *CBR*	10 (11*)	travel accounts; booking details; trip information; delivery details; transport card data; travel history
Government / Public services	*MijnOverheid*; *Amsterdam*; *IND*	3	identity details; address data; official records; government identifiers; application information
Education	*DUO*; *Studielink*	2	student identity; contact details; application details; education or financial aid information
Finance / Insurance / Payment	*ABN AMRO*; *Independer*	3	profile details; tax information; account identifiers; transaction overview; financial amounts
Total	23 measured websites	27 (28*)	–

Notes. **PostNL* parcel tracking page is excluded from the test page set in Section 5 but considered in Section 7.

due to its root-level opt-out marker that, in theory, prevents translation tools from processing the page; however, see Section 7. For accounts created for the study, we completed the account profiles using synthetic but realistic values, e.g., a home address and phone number. Pre-existing accounts (e.g., on government-related services) were prefilled with our identity information.

4.3 Leakage measurements

A test page is measured after the required login, navigation, and once the page is fully loaded. We fill forms when available before triggering translation. Requests from built-in tools are captured and exported as HAR; while Immersive Translate's traffic is routed through `mitmproxy`. We can easily isolate the traffic related to the translation service from the rest of the page's dynamic requests, by observing the destination domain.

Before translation, we manually list the sensitive values present in the measured page state, including visible text, values displayed or entered in form controls, and values in hidden DOM elements or in element attributes. Account data not present in that page state is excluded. We then search the captures offline for each listed value. We refer as a test run an instance of a translation tool on a given test page. We report on 5 translation tools × 27 test pages = 135 runs in Section 5.

For each run, we also inspect the HTTP request headers for possible metadata or identifiers that could be included. We record the number of translation requests and how segmented text is aggregated/spread across requests.

For each leaked value, we record the type of sensitive information and the element type that holds it, such as text in `span`/`div`, a value in `dd`, an `input[value]`, or option text.

Metric. The main metric is the *inclusion rate*: the proportion of available sensitive values on a page that are found in the captured translation requests:

$$\text{Inclusion rate} = \frac{\#\text{observed sensitive values}}{\#\text{available sensitive values}}.$$

We report the rate overall and per run, and the number of runs with at least one observed value.

4.4 Ethical considerations

The study used synthetic values where possible and researcher controlled accounts, so test values did not represent another person's identity, account, or financial information. Services were not tested when access required another person's account or private data. We do not perform active measurements of the translation services beyond the features they offer in the browser UI. Our tests only generate a very small load for translation service operators, akin to a single person making repeated use of the service across a few days. Thus, our experiments do not stress any service.

5 Measurement results

Below, we discuss our measurement results per tool, website category, sensitive value type, and element type. Then, we explore the structure of remote translation requests, the metadata they include, and privacy implications. Finally, we isolate HTML elements in a controlled page filled with sensitive information to study the impact of element types on the translation, independently of website behaviors. We confirmed that there are no outgoing translation requests in Firefox and exclude it from this section. We summarize the main privacy issues in Table 4.

5.1 Differences across translation tools

We found that 125 of the 135 runs contained at least one observed sensitive value. Across 1,295 tool–value pairs that could leak, 767 were observed in network requests, an overall inclusion rate of 59.2% across all tools (Figure 2). The four built-in tools' inclusion rates are similar (62–68%), while Immersive Translate's rate is significantly lower (38.6%). From our experience with this tool, this lower rate reflects the tools' ineffectiveness at translating all the text on a page, rather than any particular measure protecting sensitive values. Note that these rates do not represent the proportion of information leaks per page because test pages host varying numbers of sensitive values. Curiously, Brave's inclusion rate of sensitive values is higher than Chrome, although it is branded as "more private" than the latter [5].

Table 4: Overview of the backend URLs and main privacy-concerning behaviors

Tool	Backend URL	Ranking*	Respects `translate="no"`	Leaks domain/URL	Translates forms
Chrome	https://translate-pa.googleapis.com/v1/translateHtml	1291	Yes	**Yes**	No
Brave	https://translate.brave.com/translate_a/t	54,710	Yes	**Yes**	No
Edge	https://edge.microsoft.com/translate/translatetext	68	Yes	**Yes**	No
Opera	https://translation.opera-api.com/api/translate	137,471	Yes	**Yes**	**Depends**†
Immersive Translate	https://aigw1.immersivetranslate.com/v1/translation/tasks	354,318	**No**	No	**No**

Notes. *Ranking of the backend domain as seen in Umbrella's top domain list from April 1, 2026; †Opera translates prefilled `textareas`.

Table 5: Sensitive value inclusion by website category and translation tool. Highest scores per category are highlighted.

Website category	Comparable pages	Available sensitive values	Chrome	Brave	Edge	Opera	Immersive Translate	Overall
E-commerce / Marketplace	9	60	**86.7%**	**86.7%**	71.7%	85.0%	40.0%	74.0%
Transport / Travel	10	84	**64.3%**	**64.3%**	60.7%	58.3%	27.4%	55.0%
Government / Public services	3	51	31.4%	49.0%	49.0%	49.0%	**66.7%**	49.0%
Education	2	30	**46.7%**	**46.7%**	**46.7%**	40.0%	20.0%	40.0%
Finance / Insurance / Payment	3	34	82.4%	**94.1%**	82.4%	82.4%	38.2%	75.9%

Across all runs, the median inclusion rate is 100% for three of the four built-in tools, i.e., all sensitive values leaked in at least half of the tested pages. It is 84.6% for Opera and 38.5% for Immersive Translate. We explore these discrepancies in the rest of this section.

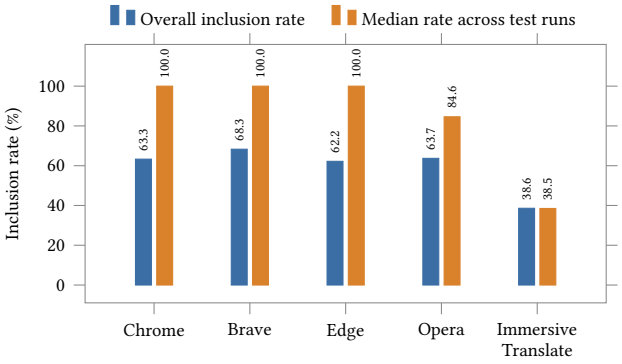


Figure 2: Observed sensitive-value inclusion by translation tool across the 135 comparable measurement runs. Bars show the overall inclusion rate over the 259 available values per tool and the median rate across the 27 test runs.

5.2 Differences across website categories

Finance/Insurance/Payment and E-commerce/Marketplace had the highest inclusion rates on average across tools (Table 5). Note that the number of websites per category and the number of sensitive values per page differ widely, as well as element types and the use of markers, so direct comparison is not fair. For instance, Finance/Insurance/Payment pages contained only 34 sensitive values, but the built-in translators included at least 82.4% of them.

Brave vs. Chrome. Brave's inclusion rates are similar or slightly higher than Chrome's. We verified that on *IND* (Finance), Brave additionally transmitted names, social security identifiers, nationality, and address text, that were not covered by opt-out markers. On *ABN AMRO*'s (Payment) transaction page, Brave additionally transmitted an account balance and the counterparty's IBAN and BIC. A possible explanation resides in the translator code that is loaded into the page upon triggering a full-page translation (as visible from the Developer Tools' Network view). Both Chrome and Brave load files from their own respective backend, which seems to correspond to different versions of the Google Translator code, with different content extraction logics.

Besides, Immersive Translate's higher rate on Government / Public services is mainly driven by *MijnOverheid*, where the tool transmitted identity and address values marked `translate="no"` that the four browsers excluded (see more details in Section 7).

5.3 Differences across sensitive value types

We distinguish 15 types of sensitive information, ranging from name, street address, city, to government identifiers and nationality. Some test pages show service-specific information, such as a list of travels (timing, starting point and destination), or an address history, which comprise multiple strings that, when taken together, reveal more information than separate strings.

Not all value types leak equally (Figure 3). We found that travel history, financial details, and email addresses leak more often than street addresses and gender, although the difference is tool-specific. For instance, Immersive Translate leaked all address histories when

built-in tools only leaked about 27%. Conversely, built-in tools leaked 66–92% of account identifiers when Immersive Translated leaked just 8.3%.

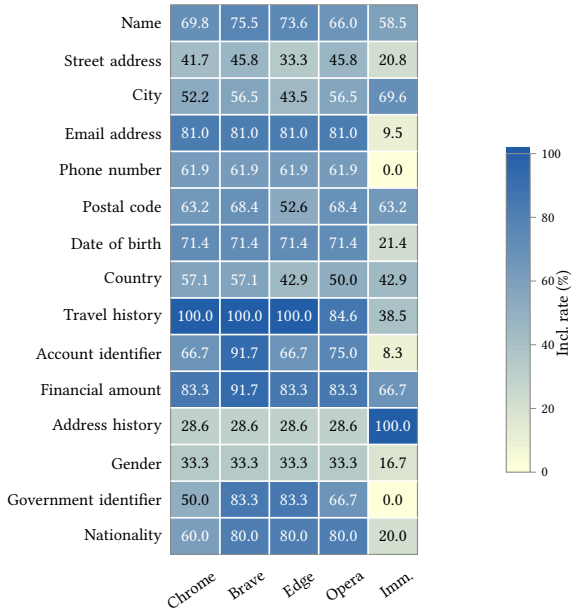


Figure 3: Observed sensitive-value inclusion rate (%) by detailed value type and tool, for value types with at least five available values per tool (Imm. = Immersive Translate).

5.4 Differences between HTML element types

Inclusion of sensitive values did not strongly depend on which HTML element they were included in (Figure 4). Most sensitive values from account cards, detail lists, order summaries, and travel history views, appear in standard text nodes such as `<div>`, `<span>`, `<p>`, which showed an inclusion rate of 74.2%. Special text nodes such as `<address>` could be treated differently due to its specific semantic; however, we found that values within this marker are actually observed more often (80%). Element attributes are also sometimes translated, e.g., `alt`, `aria-label`.

Forms. Standard form controls such as `<input>` and `<textarea>` were generally excluded. However, we distinguish the cases where such elements are prefilled from the server-side, e.g., using `<input value="...">`, or filled manually after the page is rendered. Values entered by users into input controls at runtime were not observed. However, Opera leaked a name from a prefilled textarea on *CBR*. More concerning, we found on multiple websites that the content of standard form controls is mirrored into text nodes by the web application's logic, which is then translated. Also, a form on *Marktplaats* uses a rich-text editor implemented as a `<div>` with `contenteditable="true"`, which content is fully captured for translation by all tools.

Credit card numbers. Our webpage corpus did not contain credit card numbers, so we ran a separate test on ten Dutch checkout pages: eight from websites in our main corpus, plus *IKEA* and *wehkamp* taken from the top `.nl` list. On *Airbnb* and *Amazon*, the card fields were inside a payment provider's iframes. We entered a fake but seemingly valid credit card number, expiry date, security code, and cardholder name, then translated the page. None of the entered values appeared in any request. This is consistent with the low inclusion of form data we found earlier.

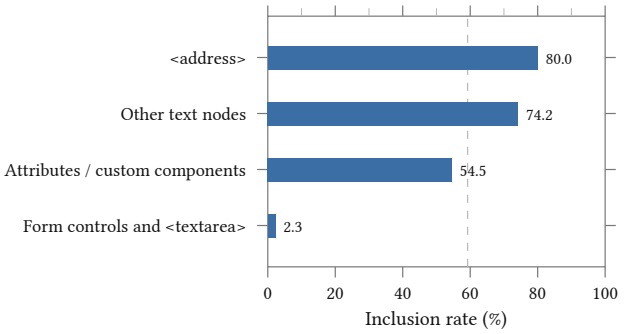


Figure 4: Observed sensitive-value inclusion by element type across the 135 comparable measurement runs, pooled over all five tools. The dashed line marks the overall rate of 59.2%.

5.5 Reference test and validation

Our website corpus covers limited HTML elements and possible page states. We build a reference page to compare translation tools using a greater diversity of testable elements and sensitive values.

We selected elements and attributes from the HTML Standard [39] and populated the page with 216 synthetic sensitive values. We tested five states: after page loading (**S0**); after entering values in input, textarea, and rich text controls (**S1**); revealing hidden content and opening a dialog (**S2**); inserting DOM content dynamically (**S3**); and after applying opt-out controls (**S4**). Note that S4 adds `translate="no"` and `class="notranslate"` to one test block only, leaving the rest of the page unchanged. Table 6 reports the observed values over all values applicable in each state.

Table 6: Observed sensitive value inclusion by browser and page state in the reference test.

Browser	S0	S1	S2	S3	S4
Chrome	**148/203 (72.9%)**	**150/210 (71.4%)**	**148/203 (72.9%)**	**152/208 (73.1%)**	**147/203 (72.4%)**
Brave	**148/203 (72.9%)**	147/210 (70.0%)	138/203 (68.0%)	141/208 (67.8%)	**147/203 (72.4%)**
Edge	121/203 (59.6%)	121/210 (57.6%)	126/203 (62.1%)	125/208 (60.1%)	120/203 (59.1%)
Opera	127/203 (62.6%)	129/210 (61.4%)	127/203 (62.6%)	131/208 (63.0%)	126/203 (62.1%)

Notes. Each cell gives observed / applicable values and the percentage. Denominators include all applicable values, including those outside opt-out regions.

```
[ [ [ ...items... ], "nl", "en" ], "te_lib" ]      // request envelope: source nl, target en
 0: [ 0 ... 99 ]                                    // array of page-text items
     0: "Ervaar hoe jouw reis met NS slimmer ..."   // interface / marketing text
     1: "Mijn NS"
     2: "Uitloggen"
     3: "overzicht ov-chipkaarten &amp; ov-chipkaart toevoegen"
     4: "Overzicht van kaartnummer: <REDACTED OV-chipkaart id>"   // sensitive value
     5: "Bekijk reishistorie"
     6: "man-is-aan-het-bellen-op-arnhem-centraal"
     7: "je reishistorie"
     ...
```

Figure 5: Example structure of a Chrome Translate request captured on NS.nl (transportation company). It carries an array of text segments mixing visible Dutch text with a OV-Chipkaart (transportation card) identifier (item 4, redacted here).

The results clearly show that some elements or configuration of elements are not translated, with noticeable differences between each tool. Filling standard and non-standard forms (S1) resulted in increased leaks, consistent with our prior observations (Section 5.4). As expected, revealing hidden (S2) or new elements (S3) typically triggered additional translations. However, Brave included 148 values in S0 but 138 in S2, despite more content being revealed. The ten missing values were existing headings, navigation text, and other page text. The cause of these omissions remains unclear.

Figure 6 combines all five states and break down the inclusion rate by group of elements. For each test group and browser, we divide the total observed count by the total applicable count across S0–S4. Each value is counted once per applicable state. Heading text and visible DOM text were frequently included. Runtime values in standard form controls were excluded by all browsers. Only Opera sent the initial text of the two prefilled textareas (one editable and one `readonly`). All four browsers sent the associated `label`.

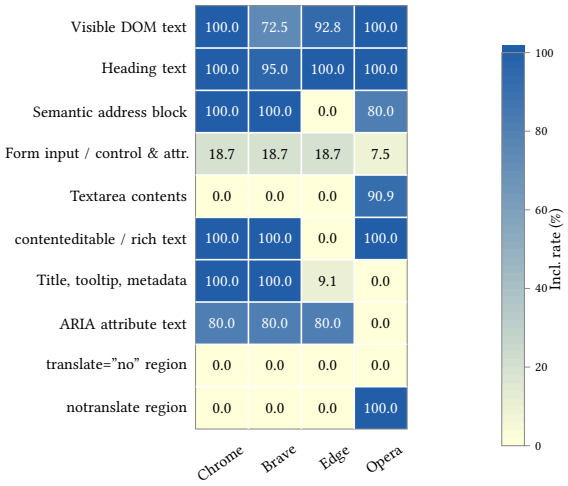


Figure 6: Observed sensitive value inclusion rate (%) by test group and browser, pooled across all five states (S0–S4). The textarea group includes its associated label.

Semantic elements (such as `address`), attribute text (such as `title` and ARIA labels), prefilled textarea content, and `contenteditable` regions showed larger differences between browsers. For instance, Edge does not translate the content in the `<address>` tag or those with the `contenteditable="true"` attribute. Such differences in design choices demonstrate that the choice of HTML element type is relevant for translation. Our black box testing method is however limited and cannot reveal all the exact rules used by each tool.

6 Request structure, identifiers & linkability

Intuitively, if a translation request includes both a sensitive item that requires translation (e.g., health-related string such as "Diagnosis: cancer") together with the user's full name, the translation service provider knows that both elements appear on the same page and is able to link both pieces of data. Better privacy can be achieved when both items are sent separately to break the relationship.

Request splitting. Translation tools split text segments across multiple requests to the translation API. We show the number of requests per tool in Table 7. A sample remote translation request from Chrome is given in Figure 5. It is composed of the source and target languages and an array of text segments. Other tools follow similar structures. We found a maximum of 304 segments per request for Chrome, 170 for Brave, 100 for Edge, and 50 for Opera, 32 for Immersive Translate. However, we found no evidence that the tools split content to separate sensitive values.

Table 7: Translation request counts in the 135 runs

Tool	Runs	Requests	Median	Min–Max
Chrome	27	45	1	1–7
Brave	27	109	3	1–18
Edge	27	185	3	1–45
Opera	27	134	4	1–19
Immersive	27	388	10	3–61
All tools	135	861	3	1–61

Domain and full URL leakage. We found that all built-in tools' requests exposed the source domain through an `Origin` header, and sometimes exposed the full URL through the `Referer` header, although this depends on the website `Referrer-Policy` header. Translation service providers can associate the text submitted to the website it originates, further increasing privacy exposure. For instance, the association of the string "*diagnosed with cancer*" with

the origin "news.com" does not carry the same meaning as if it was linked to "my.hospital.com". Browser-based translation tools therefore increase privacy leakage when compared to using online translation services where users simply copy-paste content to be translated. It is unclear whether knowing this information helps the translation or is simply an artifact of the way remote translation requests are implemented in browsers. Knowing the website on which the text is to be translated might provide more accurate translation, but we found no information to substantiate this idea.

Linking requests. Translation service providers might link requests based on a number of factors. First, the user's IP address could be cross-referenced; however, in practice, due to the widespread deployment of Network Address Translation (NAT) and Carrier-Grade NAT, tracking by IP address is unreliable [21].

Second, if all requests are sent on the same TCP/TLS connection/session, they are naturally linked. We found different behavior across tools: Chrome, Edge, and Immersive Translate reused a connection for multiple requests, while Brave and Opera used separate TCP connections and independent TLS sessions (i.e., not resumed).

Third, if requests originate from rare domains or the full URL comprises unique identifiers, the provider can use it to link requests for the same translation job. All built-in tools offer this capability.

Fourth, if a (quasi) unique identifier or cookie is shared across requests, the provider can link them, possibly across translation jobs and websites. Translation tools sometimes make use of API keys or authorization information, e.g., Immersive Translate sends the same `Authorization` header with all requests in one measurement run, but the value changes between runs. Sometimes, a cookie including a "deviceId" value was found. The Immersive Translate's privacy policy states that this value corresponds to a "pseudonymous identifier" that "is not intended to directly identify you." However, it could clearly serve as a bridge between requests and websites, and link PIIs together.

7 Opt-out marker adoption and compliance

Adoption. Four of the 28 tested pages carry opt-out markers (Table 8). In the main dataset of 27 pages, only *MijnOverheid* makes use of `translate="no"` to cover 24 values, but fails to cover the name of the user who is logged in. *TUI* and *Airbnb* only use the `notranslate` meta tag. This results in only 24/259 sensitive values being covered (9.3%). *PostNL* covers all sensitive values.

Compliance. The standard `translate="no"` attribute produced a clear split: on *MijnOverheid*, none of the 24 marked values appeared in the built-in tools' requests; however, Immersive Translate ignored the marker and included 19 of them (79.2%). On *PostNL*, built-in tools did not even issue a network request, possibly due to the document-wide `translate="no"` on the `html` tag, but Immersive Translate translated 73.3% of available sensitive values. On *Bol.com*, we discovered the custom attribute `data-no-translate` on six elements; however, it had no effect on any of the tested tools. Google/Microsoft's `meta` tag did not stop translation, as expected, and all tools translated the page and sensitive values were observed.

Table 8: Adoption and placement of translation opt-out markers on the five marked pages. Coverage counts sensitive values within marked elements or regions; meta tags cover no individual value.

Page(s)	Marker	Location	Covered / available
MijnOverheid	`translate="no"`	element & region	24 / 25
TUI	`meta notranslate`	header	0 / 15
Airbnb	`meta notranslate`	header	0 / 4
Main dataset (27 pages, 259 values)			24 / 259
PostNL	`translate="no"`, `meta notranslate`	`html` tag & header	15 / 15

8 Local Translate Guard: Marking sensitive values before translation

Faced with little countermeasures to the privacy leakages we found, we propose to empower users with a simple and automated mechanism to reduce exposure of the most easily preventable leakages. We discuss the design of the tool for Chrome, and evaluate it with Chrome's built-in translation and Immersive Translate. Both require different approaches due to the latter not respecting the `translate="no"` attribute. Finally, we discuss how the tool can be improved and ported to different browsers.

8.1 Design

We built a Chrome extension that users must trigger before translating a page to protect sensitive values, then they can translate the current page with their preferred tool. The process is depicted in Figure 7.

Sensitive value detection. The extension scans text nodes in the page DOM and detects likely sensitive values using three mechanisms: 1) regular expressions (regex) we write for values with clear formats, such as email addresses, Dutch postcodes and dates, and structured references; 2) Microsoft Presidio [19], an Natural Language Processing tool that detects PIIs such as names, locations, dates, and phone numbers; and 3) a list of known strings supplied by the user. Regex rules and custom strings are matched in the browser. When Presidio is enabled, text from HTML nodes is sent to a local Python service, which returns the type and character positions of each detected entity.

Protection. For each part of a text node that is detected as a sensitive value, we wrap it in a `span` with `translate="no"` and `class="notranslate"`. The original value remains visible, while the surrounding text remains available for translation. Input values, textareas, and editable regions are also marked as a whole. We found that built-in tools consistently respect this marking and never translate such elements.

For tools that do not honor these markings, we propose an optional *masking mode*, where the extension replaces detected values with unique placeholder tokens and keeps the original values

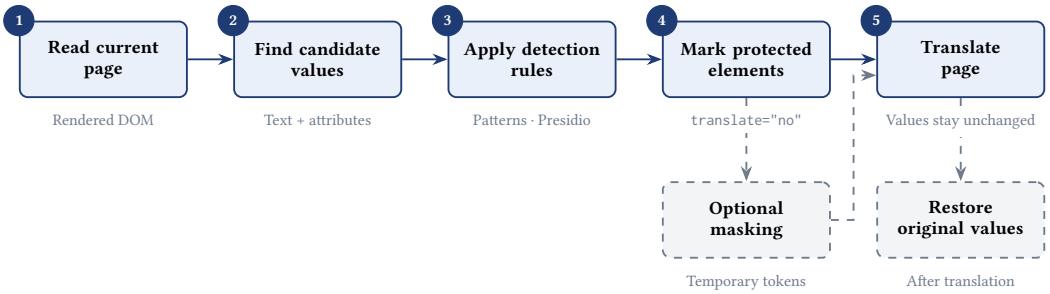


Figure 7: Workflow of Local Translate Guard. The dashed path is the optional masking mode.

in memory. After translation, the user invokes restoration to replace the tokens with the saved values. This method assumes the placeholders will not be translated or changed by the translation operation, so the original values can be swapped back.

The design follows work on sanitizing prompts in the browser, such as Casper [6], applied here to webpage translation.

8.2 Evaluation

On Chrome, we compare a baseline with two protected configurations: regex rules only, and regex plus Presidio, both using marking. Chrome is used because it respects the inserted marker (Section 7). All conditions were checked against the 259 values from Section 5; the Chrome baseline leaked 164 of them. For each value seen at baseline, we report whether it is *fully blocked* (absent), *at least reduced* (only a fragment remains, e.g. a surname without the full name), or *unchanged*. We separately evaluate marking together with masking on Immersive Translate, the tool that ignores the marker. The custom user list is not used in our experiments, to keep the evaluation focused on automatic detection.

Chrome results. Marking with regex rules lowered the inclusion rate to 41.7%, a relative reduction of 34.1% from the baseline; adding Presidio lowered it to 32.8%, a relative reduction of 48.2%. The extra protection costs time: the regex configuration has a median protection time of 9ms, the combined configuration takes 1.22s. The effect was largest on government, education, and travel pages and smallest on finance and payment pages, whose amounts and account references often lacked clear formats (Table 9). By value type (Table 10), the prototype removed structured values well (postal codes, dates of birth, email addresses) and added some coverage for names, countries, and addresses, while many city names and financial amounts remained.

Immersive Translate results. Because Immersive Translate ignores `translate="no"`, marking alone gives it no protection, so we used marking together with masking. This fully blocked 34 of the 100 baseline values and reduced five more. Protection was effective for the values we could mark and mask.

8.3 Residual exposure and improvements

With regex and Presidio on Chrome, 85 baseline values remained complete and ten remained in part. Protection depends on detecting the value before translation. Regex rules cover selected formats, leaving amounts and account references outside those patterns unmarked. Presidio adds entity recognition, but each text node is analyzed separately, limiting the context available for names and addresses. Values split across nodes may, therefore, be only partly

Table 9: Effect of the combined configuration regex + Presidio by website category in Chrome.

Website category	Baseline	Fully blocked	At least reduced
Government / Public services	16	56.3%	62.5%
Education	14	42.9%	57.1%
Transport / Travel	54	50.0%	57.4%
E-commerce / Marketplace	52	44.2%	48.1%
Finance / Insurance / Payment	28	14.3%	17.9%

Notes. Baseline is the number of sensitive values that appear in the Chrome baseline requests.

Table 10: Effect of the combined configuration by sensitive value type.

Value type	Baseline	Fully blocked	At least reduced
Name	37	24.3%	29.7%
Email address	17	70.6%	82.4%
Phone number	13	46.2%	61.5%
Travel history	13	46.2%	53.8%
Postal code	12	91.7%	91.7%
City	12	8.3%	8.3%
Street address	10	30.0%	30.0%
Date of birth	10	80.0%	90.0%
Financial amount	10	10.0%	10.0%
Account identifier	8	25.0%	25.0%
Country	8	62.5%	75.0%
Rarer types (counts: blocked / at least reduced / baseline)			
Nationality	3	2 / 2 / 3	
Government ID	3	0 / 0 / 3	
Transport card ID	3	1 / 1 / 3	
Address history	2	1 / 2 / 2	
Gender	2	1 / 1 / 2	
Transaction date	1	0 / 0 / 1	

detected. Masking uses the same detections and cannot hide values that were missed. The extension also scans the page only when requested, so content added later requires another pass. Potential improvements include a model trained for Dutch text, using nearby field labels and combining adjacent text nodes during detection, and repeating protection after page updates. Broader detection could also mark ordinary text and prevent its translation; further evaluation should therefore measure unnecessary exclusions and the effect of DOM changes on page functionality. Importantly, the user must disable the tools' automatic translation so as to first mark sensitive values with Local Translate Guard.

With better PII detection, higher protection can be achieved. Related work in the area of LLM prompt sanitization could be explored and applied here.

Portability. The protection step uses DOM operations and runs independently of the selected translation tool. We therefore expect it to require few changes on other Chromium browsers, such as Edge. A Firefox version would need adjustments to the extension manifest and background configuration, but could reuse the detection rules, DOM modifications, and local Presidio service.

9 Discussion and recommendations

Summary of results. The practical translation boundary often follows the rendered page content rather than the privacy boundary users might expect. Sensitive values were indirectly protected due to how translation tools process a page and which elements they consider out of scope for translation. Edge decides that the `address` block is out of scope, while Opera does not translate `aria-label`. No protection was implemented by the tools as a result of what they could consider sensitive. Few websites made use of opt-out markers, and sometimes missed the marking of sensitive values. A single opt-out marker was respected by all browsers, but was missed entirely by the browser extension we tested. As a result, neither opt-out markers nor client-side detection provide complete protection alone. Our proposed browser extension is able to significantly reduce the number of private leakages, but is bound to miss values due to difficulties in detecting them in unstructured text. We rely on regular expressions and an NLP tool for the task, which could be improved. Since we are mostly able to restrict the leakage of fixed-format identifiers like email addresses, we believe the remaining leaked values have less chance of being linked to some users' identifiers; however, we have not evaluated this effect.

Privacy policies and observed transmission. The commitments made by the different vendors in their privacy policy do not prevent sensitive values from reaching their service, as observed for all tools besides Firefox. For Firefox, we observed no outgoing translation requests, consistent with its documented local processing [26]. Our measurements show what is transmitted, but cannot verify the subsequent deletion or reuse promised in some policies. Apple suggests that recording domain/URL is useful to improve their service. No other vendors provide detailed explanations of their use of such information. We challenge the need for collecting the domain or full URL when translating a page.

Recommendations. Given our observations, we make the following short-term recommendations. *Users* should consider that cloud-based browser translation captures all page elements irrespective of whether they require translation or not, even non-verbal elements such as phone numbers or digital identifiers. They should therefore favor tools that offer local translation if they are concerned about exposing sensitive information to third parties. Users could also selectively translate text that does not include such values.

Web developers should mark sensitive regions with the standard `translate="no"` attribute on an element that cleanly covers them, including values shown after login or inserted by scripts. However, they cannot guarantee that the translation tool chosen by the user will respect this choice. However, see our discussion regarding responsibilities below.

Browser and extension vendors must respect the standard `translate` marker and its inheritance, and should consider implementing filters to avoid sending obvious invariable strings to their translation service. We are concerned that we found no filter based on the text content implemented in any of the tested tools. Moreover, vendors should stop collecting domain and URL information as part of calls to the translation API, as this does not appear to be of direct benefit to the user. Furthermore, as demonstrated by our developed extension Local Translate Guard, translation tools could perform basic checks to avoid leaking the most obvious sensitive values. They could also split text segments across as many API calls/network requests as possible to avoid linking them server-side. However, this measure would carry a negative performance impact. Finally, when practical, these vendors should favor local translation. As of the time of this publication, Chrome supports a local AI-based Translation API [7] for website developers, but does not offer it for full-page translation triggered by the user.

Whose responsibility it is to protect sensitive data. Despite the above recommendations, the issue of privacy exposure during translation does not point to a single responsible party. While the user agrees to use the translation service and should understand the privacy implications, automated translation may bring surprises, and the user has no easy way to translate only some of the content to avoid sending e.g., identifiers. Due to how browser translation works, pages could sometimes be automatically translated, contributing to leakages before the user can even realize.

Moreover, the `translate` attribute is not primarily intended for developers to mark sensitive values and avoid translation *for privacy reasons* by assuming that privacy leakages would occur otherwise. In Firefox, for instance, no leakage occurs during translation due to the local translation engine. The `translate` attribute is rather reserved for cases where translation would distort the message.Another option would be to standardize HTML markup to label sensitive values that should be protected by more privacy-focused processing, e.g., akin of the `translate` attribute, there could be a `private` attribute.

Importantly, other information could be deemed sensitive other than the values we considered. The theme of a page (e.g., reading about a medical condition) already leaks the user's interests. Therefore, a more generalized solution to privacy concerns during

automatic translation is to enable on-device translation. Recent advancements in LLMs and computing power of user devices should ultimately enable such an evolution.

Limitations. Our corpus of 23 Dutch websites across various categories provides a good idea of the privacy issues on Dutch services but is not necessarily representative of all of them or of services in other languages. Especially healthcare and banking websites are underrepresented because they often require an existing account that involves registration using a real identity or even physical registration. This barrier is common to any study of authenticated pages. Although we investigated 5 major browsers, we could not evaluate Safari due to technical limitations that prevent us from inspecting its translation-related requests. Furthermore, the black-box nature of our study identifies values that appear in captures but cannot inspect internal processing; a negative observation is not proof a value was never processed or would not in slightly different conditions. The reference page we build to study the tools' behaviors does not exhaustively enumerate all possible ways in which content can appear on a page. In particular, we saw that forms can be implemented in non-standard ways on some websites, which resulted in privacy leakages despite standard forms being generally spared from translation.

Reproducibility and test automation. Due to the use of our own personal accounts in most experiments, we are unable to share network traces with the community.

Our study perform manual tests of Dutch websites. Although the process of browsing and collecting network traces could be automated, the main barrier to scaling our tests is the necessity to have (meaningful) accounts provisioned at the identified services. Indeed, unlike other web measurement studies, sensitive information is not displayed, by definition, on public pages that can be easily crawled. Instead, only private, login-protected pages are interesting to study. The most sensitive values are being found on services bound to the user's real identity, for which synthetic accounts cannot be created. An automated analysis would register accounts and provision them with unique placeholder data that fulfills the expected format, then trigger translation of all pages to compare the available placeholder data in the DOM with what appears in network requests to translation providers. Studying the reasons for values to be leaked could be more challenging without studying the source code of the tools.

10 Conclusion

We highlight the extent of the privacy issues behind browser-based webpage translation in a multi-tool multi-website study of authenticated pages. We show that full-page translation is often implemented without privacy considerations by most major browser vendors and the tested Chrome extension vendor to avoid leaking sensitive values to cloud-based translation services. With the exception of Firefox since 2023 with Project Bergamot, other major browsers consistently leak nearly all content in text nodes. Forms are treated in a special way, with user inputs being systematically excluded. However, websites may implement custom forms or create copies of form contents into regular text nodes, exacerbating the issue. Although our dataset is small, translation tools clearly do not sufficiently protect sensitive data that require no translation, and website operators leverage opt-out markers insufficiently. We even discovered a systematic and unnecessary leakage of the origin domain and sometimes the full URL to the translation service, with Apple storing this data for up to five years according to its privacy policy. Future work could expand our study to quantify privacy exposure at scale; however, automation is made difficult by the need to provision real accounts, and we believe solutions to the privacy issue are more important to focus on.

Opt-out markers gave incomplete protection because they were rare and handled inconsistently: in our tests only the standard `translate="no"` attribute worked, and only for the built-in tools. We proposed a browser extension that applies simple regex in combination with a small NLP-based tool to easily protect values before translation. While this approach can provide significant protection for certain value types and provide immediate control over translation for users, we open the debate about who is responsible for fixing this privacy problem. Web developers are not supposed to prevent translation of sensitive values, as a potential leakage depends on the browser.

Improving the detection of PII/sensitive values by leveraging PII detection as typically applied in other fields should remain limited to lightweight methods; otherwise, the cost of PII detection may outrun that of simply translating a page locally. To avoid any privacy exposure to third parties, local/on-device translation mechanisms seem more effective.

Acknowledgments

We are grateful to anonymous reviewers for their comments and suggestions. We also thank Gunes Acar for his support and valuable inputs.

References

[1] Sultan Alneyadi, Elankayer Sithirasenan, and Vallipuram Muthukkumarasamy. 2016. A Survey on Data Leakage Prevention Systems. *Journal of Network and Computer Applications* 62 (2016), 137–152. doi:10.1016/j.jnca.2016.01.008

[2] Apple. 2026. Webpage Translation in Safari on Mac. https://support.apple.com/en-us/guide/safari/ibrw6ea421e3/mac. Accessed: 2026-03-31.

[3] Nikolay Bogoychev, Jelmer Van der Linde, and Kenneth Heafield. 2021. Translate-Locally: Blazing-fast translation running on the local CPU. In *Conference on Empirical Methods in Natural Language Processing: System Demonstrations*. Association for Computational Linguistics, 168–174. doi:10.18653/v1/2021.emnlp-demo.20

[4] Brave. 2026. Brave Browser Privacy Policy. https://brave.com/privacy/browser/. Accessed: 2026-03-31.

[5] Brave Software. [n. d.]. Brave vs Chrome. https://brave.com/compare/chrome-vs-brave/. Accessed: September 12, 2026.

[6] Chun Jie Chong, Chenxi Hou, Zhihao Yao, and Seyed Mohammadjavad Seyed Talebi. 2025. Casper: Prompt Sanitization for Protecting User Privacy in Web-Based Large Language Models. In *Conference on Cyber Security and Cloud Computing (CSCloud)*. 122–133. doi:10.1109/cscloud66326.2025.00027

[7] Chrome for Developers. 2025. Translation with built-in AI (Translator API). https://developer.chrome.com/docs/ai/translator-api. Accessed: 2026-03-07.

[8] Aldo Cortesi, Maximilian Hils, and Thomas Kriechbaumer. 2026. *mitmproxy: An interactive TLS-capable intercepting HTTP proxy.* https://mitmproxy.org

[9] Kelly Davis. 2019. *Bergamot Deliverable D6.1: Basic Firefox Integration.* Technical Report. Bergamot Project (Horizon 2020 RIA), Grant Agreement No. 825303. Date of Submission: 2019-09-30. https://browser.mt/assets/D6.1.pdf

[10] Google. [n. d.]. Google Terms of Service. https://policies.google.com/terms. Accessed: July 17, 2026.

[11] Google. 2026. Translate pages and change Chrome languages. https://support.google.com/chrome/answer/173424?hl=en. Google Chrome Help. Last Accessed: July 17, 2026.

[12] Google. 2026. *Translate pages and change Chrome languages - Computer - Google Chrome Help.* https://support.google.com/chrome/answer/173424?hl=en

[13] Immersive Translate. 2026. Immersive Translate. Accessed: 2026-04-23. https://immersivetranslate.com/en/

[14] Immersive Translate. 2026. Immersive Translate Privacy Policy. Accessed: 2026-04-23. https://immersivetranslate.com/docs/PRIVACY/

[15] Paweł Kamocki, Jim O'Regan, and Marc Stauch. 2016. All Your Data Are Belong to us. European Perspectives on Privacy Issues in 'Free' Online Machine Translation Services. In *Privacy and Identity Management*. Edinburgh, UK. doi:10.1007/978-3-319-41763-9_18

[16] Chaejin Lim, Beomjin Jin, and Hyoungshick Kim. 2025. Stealth Extension Exfiltration (SEE) Attacks: Stealing User Data without Permissions via Browser Extensions. In *ACM/SIGAPP Symposium on Applied Computing (SAC'25)*. Catania, Italy, 1820–1828. doi:10.1145/3672608.3707856

[17] MDN Contributors. 2025. HTML global attribute: `translate`. https://developer.mozilla.org/en-US/docs/Web/HTML/Reference/Global_attributes/translate. Accessed: 2026-03-08.

[18] Microsoft. 2026. No Trace - Microsoft Translator for Business. https://www.microsoft.com/en-us/translator/business/notrace/. Accessed: 2026-03-31.

[19] Microsoft. 2026. *Presidio: Data Protection and De-identification SDK*. Accessed: 2026-05-18.

[20] Microsoft. 2026. *User data and privacy in Microsoft Edge*. Section "Translate". https://learn.microsoft.com/en-us/legal/microsoft-edge/privacy

[21] Vikas Mishra, Pierre Laperdrix, Antoine Vastel, Walter Rudametkin, Romain Rouvoy, and Martin Lopatka. 2020. Don't Count Me Out: On the Relevance of IP Address in the Tracking Ecosystem. In *The Web Conference (WWW'20)*. 808–815. doi:10.1145/3366423.3380161

[22] Mozilla. [n. d.]. Overview of Translations. https://firefox-source-docs.mozilla.org/toolkit/components/translations/resources/01_overview.html. Accessed: 2026-05-10.

[23] Mozilla. 2022. Mozilla releases local machine translation tools as part of Project Bergamot. https://blog.mozilla.org/en/mozilla/local-translation-add-on-project-bergamot/. Last Accessed: July 17, 2026.

[24] Mozilla. 2023. Firefox 118.0 Release Notes. https://www.mozilla.org/en-US/firefox/118.0/releasenotes/. September 26, 2023. Accessed: September 12, 2026.

[25] Mozilla. 2024. Firefox Translations Add-on. https://support.mozilla.org/en-US/kb/firefox-translations-add-on. Mozilla Support. Last Accessed: July 17, 2026.

[26] Mozilla. 2026. Firefox Translations - Mozilla Support. https://support.mozilla.org/en-US/kb/website-translation. Accessed: 2026-03-31.

[27] Opera. 2026. Opera Translate | Built-in Webpage Translation. https://www.opera.com/features/translation. Accessed: 2026-03-23.

[28] Project Bergamot Consortium. 2022. Bergamot – a project to add and improve client-side machine translation in a web browser. https://browser.mt/. EU Horizon 2020 Research and Innovation Programme, grant agreement No. 825303..

[29] Ryoma Sato. 2023. Making Translators Privacy-aware on the User's Side. arXiv:2312.04068 [cs.CR] https://arxiv.org/abs/2312.04068

[30] Asuman Senol, Gunes Acar, Mathias Humbert, and Frederik J. Zuiderveen Borgesius. 2022. Leaky Forms: A Study of Email and Password Exfiltration Before Form Submission. In *USENIX Security Symposium*. Boston, MA, USA, 1813–1830.

[31] Similarweb. 2026. Top Websites Ranking in the Netherlands. https://www.similarweb.com/top-websites/netherlands/. Accessed: 2026-04-12.

[32] Oleksii Starov and Nick Nikiforakis. 2017. Extended Tracking Powers: Measuring the Privacy Diffusion Enabled by Browser Extensions. In *World Wide Web (WWW'17)*. Perth, Australia, 1481–1490. doi:10.1145/3038912.3052596

[33] StatCounter Global Stats. 2026. Desktop Browser Market Share Worldwide. https://gs.statcounter.com/browser-market-share/desktop/worldwide. Accessed: 2026-03-15.

[34] The Chromium Project. [n. d.]. Translate (Chromium Design Document). https://www.chromium.org/developers/design-documents/translate/. Accessed: 2026-03-07.

[35] Tranco. 2026. Tranco List. https://tranco-list.eu/. Accessed: 2026-04-12.

[36] Christine Utz, Sabrina Amft, Martin Degeling, Thorsten Holz, Sascha Fahl, and Florian Schaub. 2023. Privacy Rarely Considered: Exploring Considerations in the Adoption of Third-Party Services by Websites. *Proceedings on Privacy Enhancing Technologies (PoPETs)* 2023, 1 (2023), 5–28. doi:10.56553/popets-2023-0002

[37] Lucas Nunes Vieira, Carol O'Sullivan, Xiaochun Zhang, and Minako O'Hagan. 2023. Privacy and everyday users of machine translation. *Translation Spaces* 12, 1 (2023), 21–44. doi:10.1075/ts.22012.nun

[38] W3C Internationalization Working Group. 2014. Using HTML's `translate` attribute. https://www.w3.org/International/questions/qa-translate-flag.en.html. Accessed: 2026-03-04.

[39] WHATWG. 2026. HTML Standard: Index of Elements. https://html.spec.whatwg.org/multipage/indices.html. Living Standard, accessed: 2026-05-06.

[40] Qinge Xie, Manoj Vignesh Kasi Murali, Paul Pearce, and Frank Li. 2024. Arcanum: Detecting and Evaluating the Privacy Risks of Browser Extensions on Web Pages and Web Content. In *USENIX Security Symposium*. Philadelphia, PA, USA, 4607–4624.