TypoAlert: a browser extension against typosquatting

Francesco Blefari[1,2,*], Angelo Furfaro[1], Giovambattista Ianni[1] and Alessandro Viscomi

[1] University of Calabria, Rende (CS), Italy
[2] IMT Schools for Advanced Studies, Lucca (LU), Italy

Abstract

Nowadays, web browsing has become ubiquitous, with virtually everyone navigating the internet and routinely entering website addresses. However, frequent typing can lead to errors, resulting in the inadvertent input of incorrect domain names. One prevalent risk stemming from such mistakes is known as typosquatting, where users inadvertently land on maliciously crafted websites due to minor typing errors. By exploiting typographical errors made by users, typosquatting represents a malicious tactic wherein attackers capitalize on such mistakes to redirect unwitting victims to entirely different or deceptively similar websites. While various techniques and tools have been developed to mitigate this threat, currently, there is a notable absence of user-friendly tools available to everyday web users. This paper describes TypoAlert, a Chrome-based extension engineered to address this gap in defense against typosquatting. TypoAlert is meticulously crafted to analyze, detect, and promptly alert users in real-time about the legitimacy of the web domains they are visiting.

Keywords

TypoSquatting, URL Hijacking, Privacy, Phishing

1. Introduction

In the current panorama of digital threats, cybersquatting represents an illicit activity aimed to hijack domain names that correspond to trademarks or famous personalities. Over time, this threat has evolved into the phenomenon known as *typosquatting*: a threat based on typing errors made by users when entering a URL into their browser. The attackers, called (*typosquatters*), register domains that contain spelling errors compared to legitimate domains, taking advantage of people's inevitable oversights. This kind of attack is particularly effective when the reference domain is frequently visited because even a small percentage of user typing errors generates a significant flow of traffic to *typosquatted* sites.

Typosquatted sites are web sites whose domain name are similar to legitimate domain name, and can host a wide range of content aimed to generate profits through advertising and often containing malicious elements and/or redirects to malicious websites. Usually, the attackers exploit typosquatted sites to conduct attack campaigns, such as phishing, or even to steal sensitive user information. Prior research [1] indicates that a considerable percentage, ranging from 10% to 20%, of manually entered URLs contain errors. For instance, an average user who

SEBD 2024: 32nd Symposium on Advanced Database Systems, June 23-26, 2024, Villasimius, Sardinia, Italy

[*] Corresponding author.

francesco.blefari@unical.it (F. Blefari); angelo.furfaro@unical.it (A. Furfaro); ianni@unical.it (G. Ianni); a.viscomi00@gmail.com (A. Viscomi)
https://blefari.xyz/ (F. Blefari); https://angelo.furfaro.dimes.unical.it/ (A. Furfaro); https://www.mat.unical.it/ianni/ (G. Ianni)
0009-0000-2625-631X (F. Blefari); 0000-0003-2537-8918 (A. Furfaro); 0000-0003-0534-6425 (G. Ianni)

erroneously inputs the URL of a popular website has a 1 out of 14 probability of landing on a typosquatted domain [1]. The consequences of typosquatting are profound and far-reaching. Companies suffer not only from traffic declines but also from subsequent financial losses, while users remain persistently vulnerable to potential online scams. Despite the wealth of studies conducted in this domain [2], and the presentation of several prototype anti-typosquatting tools in the past, there exists a lack of practical and effective solutions available to the general public, particularly those offering real-time assistance to users. This paper presents the TypoAlert extension for chrome-based browsers and shows how combining multiple anti-typosquatting methods into an integrated framework is possible in order to implement an effective detection tool against typosquatting. Remarkably, as our methods are not machine learning-based, they do not require cycles of training on input datasets; thus the maintenance effort of TypoAlert is reduced to the bare minimum. Moreover, we assessed the effectiveness of TypoAlert on an appropriate set of domain names. Section 2 summarizes some previous studies on typosquatting and highlighting the necessary background information. Section 3 briefly presents our typosquatting detection methodology and illustrates the experimental results obtained during the evaluation and validation phase. Section 4 presents the chrome-based extension, its features and how it works. Section 5 draws the conclusions and indicates some future research directions.

2. Background and related work

The general term cyber-typosquatting might refer not just to domain name typosquatting but also to package typosquatting [3] and to other forms of typosquatting, like exploitation of typing errors in mobile app names, social media names, etc. Typosquatting remains a widespread and persistent practice, primarily due to the lack of effective solutions to prevent it [1]. Research on the topic can be roughly categorized in: *(i)* general analyses; *(ii)* company-centric anti-typosquatting proposals; and, *(iii)* user-centric anti-typosquatting research.

General studies. The study presented in [4] identified over 8800 registered domains within typographic variations compared to popular domain names and more than 90% of these redirected to sexually explicit content often designed to make it hard to shut down the offending content.

Initially, it was believed that shorter URLs were more susceptible to typosquatting [5], however, [6] indicated that domains with longer names share a similar probability of being subject to typosquatting.

Similarly, the popularity of domain names was originally seen as a factor related to typosquatting [5]. This assumption has also been revisited; indeed, a shift in typosquatters' behavior has been identified in [7]: around 95% of typosquatted domains now targets lesser popular domains.

During the years, various models for generating typosquatted domains have been proposed. Five primary models have been identified [8]: Missing-dot typos, Character-omission typos, Character-permutation typos, Character-substitution typos, and Character-duplication typos. A subsequent study [9] scrutinizes registered domains whose name has been generated according to each of the five described models, evaluating their saturation. This work also provided valuable insights into the level of awareness regarding various typo domain generation models among distinct online entities. As shown in [9], both malicious and defensive registrations mirror the saturation trends. This implies that both attackers and defenders share a similar perception regarding the typosquatted domains deemed worthy of registration.

Existing models were extended by introducing additional approaches in [10]. These include: *(i)* 1-mod-inplace: involves in replacing all domain name characters, one at a time, with every possible letter of the alphabet; *(ii)* 1-mod-deflate: entails removing, one at a time, all characters from the domain name; *(iii)* 1-mod-inflate: involves adding a character to the domain name, systematically considering all possible characters.

All these generation models are based on the Levenshtein distance (also known as edit distance) [11, 12]. This metric allows to quantify the similarity between two strings, and is thus a crucial parameter for evaluating the similarity of typo domains to the original domain. However, when considering the character permutation generation model, the more appropriate reference is the Damerau-Levenshtein distance [13] which differs from plain Levenshtein distance by incorporating the operation of transposition between characters, in addition to insertion, deletion, and substitution operations. In [2] it is highlighted that 99% of typosquatted sites exhibit a Damerau-Levenshtein distance of one from their target domains.

Company and user-centric anti-typosquatting tools. The pioneer Strider Typo-Patrol tool [8] is meant for discovering large typosquatting campaigns. It employed a multifaceted approach, incorporating *(i)* a Typo-Neighborhood Generator to produce sets of URLs with potential typos, *(ii)* a Typo-Neighborhood Scanner to actively analyze domains and record information such as third-party URLs and page content, and *(iii)* a Domain-Parking Analyzer for in-depth analysis of typosquatted domains. The same work proposed Strider URL Tracer, an instrument meant to allow website owners to monitor typosquatted domains targeting their sites. A comprehensive and relatively recent analysis of typosquatting domain registrations within the `.com` TLD can be found in [7]. The analysis was conducted using the Yet Another Typosquatting Tool (YATT).

Another approach was provided in the now defunct iTrustPage Firefox extension [14], which provided automated identification of legitimate web pages, utilizing user input and external sources such as search engine results, including whitelists and local caches. A browser extension called The Anti Typosquatting Tool (ATST) was proposed in [1]. It provided several features such as: *(i)* a User Customized Local Repository for monitoring popular domains, *(ii)* an Edit-distance Computation Module employing the Damerau-Levenshtein distance for typosquatted domain checks, and *(iii)* a User Customized Local Repository Update Module for dynamic updates based on user interactions. The Stop URL Typo-squatting (SUT) approach, proposed in [5], addresses the broader issue of detecting *phony* websites, whose domain name is not necessarily typosquatted. This solution integrates autonomous modules for: *(i)* network-level criteria that assesses URL features (called SUT-net module) *(ii)* and site popularity assessment that leverages Google search results to evaluate domain legitimacy (called SUT-pop module).

Another tool that is also worth mentioning is TypoWriter [15], which anticipates most likely domain variations using Recurrent Neural Networks trained on DNS logs. Unfortunately, at the time of writing, all the above mentioned tools are no longer available on the web.

3. Detection Methodology

Behind our developed Chrome-based extension there is a detection algorithm that aims to classify the type of the web domain at hand. Our tool aims to integrate several anti-typosquatting techniques and to provide real-time monitoring, detection and filtering software. We also included additional detection features meant identifying as detection of domain names which are registered, yet are not used and/or intended for bad uses called *parked domains*. The detection process starts taking in input a domain name n and, after analysis, n is classified. The output is one of the following categories: *NotTypo* (n is not a typosquat), *ProbablyNotTypo*, *ProbablyTypo*, *Typo*, *ProbablyTypoPhishing*, *TypoPhishing*, *TypoMalware*, and is built according to a score (*av*, alert value) and to the value of a phishing indicator (*ph*) which are both obtained as outcomes of the evaluation step.

A pre-filtering step is achieved by considering two lists: a blacklist (*BL*) and a whitelist (*WL*). The blacklist leverages the BlackBook list, an historical (black)list of malicious domains created as part of the periodic automated heuristic check (i.e. WHOIS, HTTP, etc.) of newly reported entries from public lists of malicious URLs [16]. The BlackBook blacklist is used to check whether the domain at hand is considered malware; if so, the domain is marked as a $TypoMalware$ domain.

Let vd be the domain name eventually reached from n after following a potential chain of redirects. If $n \in WL$ or $vd \in WL$, we give to n the minimum alert value, i.e. 0, classifying it as *NotTypo*. The *WL* list is constructed using a *Top Domain Repository* (*TDR*), giving at the same time the capability of adding more domains using the *User Domain Repository* (*UDR*); this latter can be populated directly by using the web page related to the developed extension. The *WL* is a list that can be reasonably assumed to be reliable and authentic built considering the top domains provided by Data4Seo [17]. Data4Seo website allows to export data concerning the top 1000 national web domains for each of the 74 distinct nations available and also the 1000 web domains with the highest ranking worldwide. We added to WL all top domains present on Data4Seo website (for a total of around 32000 distinct domain names) and the user added trusted domains. TDR cannot be modified by the user, which can however customize the complementary UDR, which is initially empty.

Afterwards, we build a set CT of *candidate targets*. CT is built by considering each element having DL-distance equals to 1 from n taken from: *(i)* WL, *(ii)* the top 10 domain names resulting by querying a search engine with n as the search keyword and whenever available, *(iii)* the domain name dym (*"Did you mean?"* domain), i.e. the domain name suggested by the search engine at hand as the inferred correct search keyword.

Once the CT list is built, the evaluation step starts by computing the Parking Alert (PARKA) indicator which is set to either 0 or 1 according to an analysis based on a set of keyphrases, in different speaking languages, usually present in parked web pages. Then for each element $ct \in CT$ we evaluate the *Top 10 Alert (T10A)* indicator, the *Did You Mean Alert (DYMA)* indicator and the *Phishing Alert (PHA)* indicator.

The *T10A* indicator considers the result list obtained by querying the input domain n on a search engine; we compute the $T10A_{ct}$ score. This indicator returns: *(i)* 1 if n is present in the resulting list; *(ii)* -1 if n is not present in the resulting list; *(iii)* 0 in all other cases.

The DYMA indicator is based on the concept of *domain popularity*, and it exploits the suggested

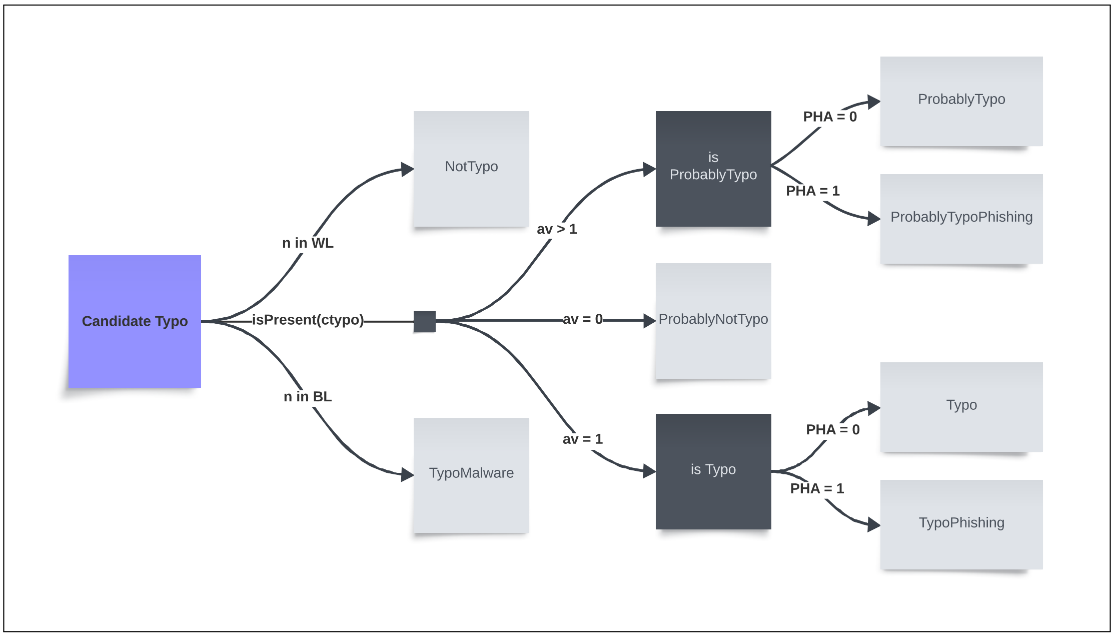


Figure 1: Labelling Algorithm

sites coming from a search engine about possible typing errors in n. Similarly to the previous indicator, it returns a score that we call DYMA$_{ct}$ that is set to 1 if n triggers the suggestion of ct in the search engine and is set to 0 otherwise.

Last but not least, there is the PHA_{ct} indicator that evaluates the similarity degree between the web page related to the input domain n and the web page related to ct. This evaluation is carried out using fuzzy hashing [18] and returns the score value 0 or 1.

Based on the above indicators, the alert value av is computed as follows:

$$av = \begin{cases} 0 & \text{if } n \in WL \\ 7 & \text{if } n \in BL \\ 2 + PARKA + av_{|CT} & \text{otherwise} \end{cases}$$

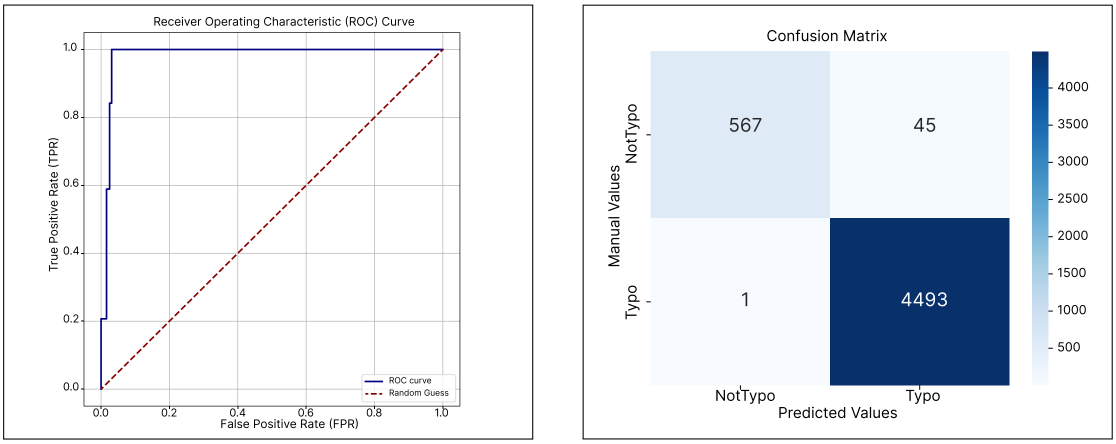
where $av_{|CT} = \max_{ct \in CT} \{(T10A_{ct} + DYMA_{ct} + PHA_{ct})\}$.

Along with av we obtain the phishing alert (ph) value as PHA_{ct^*} where ct^* is one of the arguments for which $av_{|CT}$ is reached and for which PHA is maximal, i.e.

$$ct^* = arg \max_{ct \in CT} \{PHA(ct) | T10A_{ct} + DYMA_{ct} + PHA_{ct} = av_{|CT}\}.$$

Finally, in the last step (see Figure 1) we label n according to av and ph: for $av = 0$, we assign the label $NotTypo$; for $av = 1$, we assign the label $ProbablyNotTypo$; for $av = 2$ we assign either the label $ProbablyTypo$ or $ProbablyTypoPhishing$ depending on the value of ph, respectively if 0 or 1; for $av = 7$ we assign the label $TypoMalware$ while for any value $av \in [3, 6]$ we assign either $TypoPhishing$ if $ph = 1$ or $Typo$ if $ph = 0$.

To assess the effectiveness of the classification techniques which TypoAlert is based on, we conducted an evaluation utilizing a purposely constructed dataset, named TS, including a set



(a) ROC curve

(b) Confusion matrix

Figure 2:

of potential typosquatted domains. To build the ground truth, each domain $d \in TS$ has been manually analyzed and classified as being or not a typosquatted domain. Then we compared the results with the outcomes achieved by our classifier.

To build the TS dataset we started from the set Top, comprising the top 1000 websites globally ranked on Google, as per DataForSEO [17]. We extracted a subset of 300 domains by uniformly sampling Top and using the open source tool *ail-typo-squatting* [19], we built a set containing all domain names having a Damerau-Levenshtein distance from d's name which is equal to 1. Then we extracted a subset of all domain names d such that *(i)* d was actually registered in a DNS at the time of construction of the dataset; and *(ii)* there was an active web server responding (directly or indirectly) to HTTP(S) requests made to d. Finally, we obtained TS . The final dataset *TS* includes potential 5106 typo domains.

During the evaluation phase we conducted an analysis on the tool accuracy and we compared it with ground truth obtained manually. During the manual classification we labelled domains as *(i) Typo*: designated for domains considered malicious; *(ii) NotTypo*: assigned to either a legitimate domain or a domain that redirects to the legitimate domain. Note that, to mitigate the role of human subjectivity in manual annotations, we opted for building binary ground truth values. However, since TypoAlert produces a score value between 0 and 7, data have been validated by mapping our scores to ground truth. We consider an aggregation threshold t, and we build a family of binary classifiers each denoted by the two classes $NotTypo_t = \{x \in TS \mid s(x) < t\}$, and $Typo_t = TS \setminus Typo_t$. We identified the classifier that maximizes the TPR/FPR Ratio (True positive rate divided by False Positive Rate), as the one obtained for $t = 2$. The Receiver Operating Characteristic (ROC) curve shows the trade-off between True Positive Rate and False Positive Rate of each classifier built among various score thresholds, as depicted in Figure 2a. Figure 2b depicts the confusion matrix for $t = 2$, where 5060 over 5106 domains with a 99.0% of domains were correctly classified.

4. The extension

We took several design choices in developing TypoAlert. First things first, as our software would be a browser extension, we have chosen to support all Chrome-based browsers.

TypoAlert aims to improve the user experience in browsing the web without being pervasive for the users. To carry out this goal, TypoAlert, once installed in the browser, shows as the only visible additional feature, an icon in the dedicated extension section. This icon changes its color based on the web site present on the active tab. These colors have been chosen to give users a rapid evaluation measure of the domain kind they are visiting and may vary according to the Figure 3. Given a domain name n the TypoAlert icon can assume a different color: *(i)* Blue: if the analysis is not started yet; *(ii)* Dark-Red: if n is marked as *TypoMalware* or *TypoPhishing*; *(iii)* Red: n is marked as *Typo*; *(iv)* Yellow: if n is marked as *ProbablyTypo* or *ProbablyTypoPhishing*; *(v)* Green-Yellow: if n is marked as *ProbablyNotTypo*; *(vi)* Green: if n is marked as *NotTypo*.

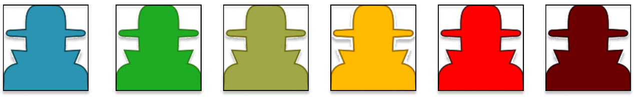


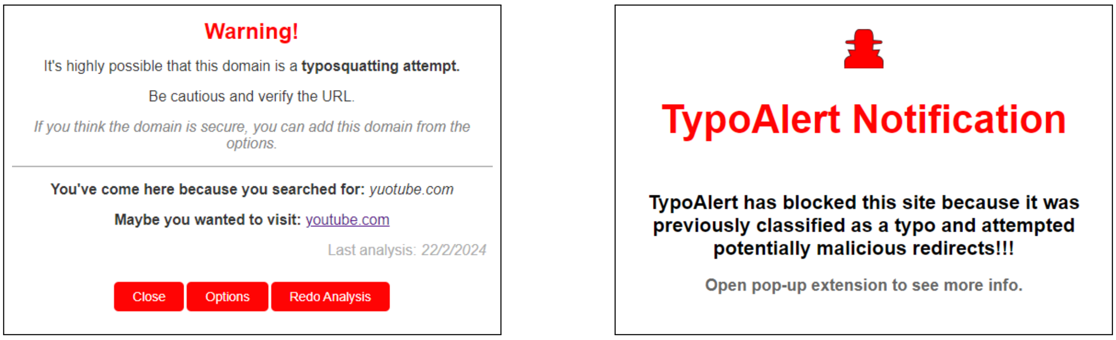
Figure 3: Different colours of the toggle extension.

Different result values returned by analysis involve different (or none) alert notification. If the extension's analysis indicates a label among *ProbablyTypo*, *Typo*, *TypoPhishing*, *ProbablytypoPhishing* or *TypoMalware* (colors from yellow to dark red), an alert appears, warning the user about the detected severity level. When the analysis returns $NotTypo$ or $ProbablyNotTypo$ no alert is given and the user is allowed to visit the related web page, in this case the TypoAlert icon becomes either Green or Green-Yellow.

If a typosquatting attempt is detected, the extension's icon becomes red and an alert about the domain classification is shown. In Fig. 4a it is depicted the alert that appears when the domain n is a typo and it is visited for the first time. It was highlighted before that the *Phishing Alert* indicator evaluates if a web domain is malicious and aims to conduct a phishing attack. If the *Phishing Alert* indicates that a web domain is a possible phishing web domain, the user is notified using a specific pop-up alert highlighting this special kind (malicious) of the web domain. Moreover, we inserted in the extension a caching mechanism that helps in avoiding multiple evaluations about the same site. Domain names classified as typosquatted are retained in the extension cache. If a domain name b has been classified as typosquatted in the last 30 days the web page of b is blocked and replaced by a notification page as depicted in Fig. 4b. Users can always access the extensions options and add misclassified domains to the verified user whitelist, excluding them from the analysis.

5. Conclusions and future work

In this paper we presented TypoAlert, a tool for detecting typosquatted sites that, combining some of the known simplest yet provably effective practices, is able to detect a relevant number of



Warning!

It's highly possible that this domain is a **typosquatting attempt.**

Be cautious and verify the URL.

If you think the domain is secure, you can add this domain from the options.

You've come here because you searched for: *yuotube.com*

Maybe you wanted to visit: youtube.com

Last analysis: *22/2/2024*

TypoAlert Notification

TypoAlert has blocked this site because it was previously classified as a typo and attempted potentially malicious redirects!!!

Open pop-up extension to see more info.

(a) Alert popup for a typosquat domain

(b) Notification page for a typosquat domain

Figure 4: TypoAlert notification.

typosquatted domains. The validation phase proves the effectiveness of the approach. As future work, we are planning to enrich TypoAlert with features that tackle typosquatting from an even more user-centric perspective, in the spirit of dynamic skins [20]. TypoAlert hase been released under LGPL license and it can be downloaded from https://github.com/aleviscomi/typoalert.

Acknowledgments

This work was partially supported by projects SERICS (PE00000014) and FAIR (PE0000013) under the MUR National Recovery and Resilience Plan funded by the European Union - NextGenerationEU.

References

[1] G. Chen, M. F. Johnson, P. R. Marupally, N. K. Singireddy, X. Yin, V. Paruchuri, Combating typo-squatting for safer browsing, in: 2009 International Conference on Advanced Information Networking and Applications Workshops, 2009, pp. 31–36. doi:10.1109/WAINA.2009.98.

[2] J. Spaulding, S. Upadhyaya, A. Mohaisen, The landscape of domain name typosquatting: Techniques and countermeasures, in: 2016 11th International Conference on Availability, Reliability and Security (ARES), 2016, pp. 284–289. doi:10.1109/ARES.2016.84.

[3] M. Taylor, R. Vaidya, D. Davidson, L. De Carli, V. Rastogi, Defending against package typosquatting, in: Network and System Security: 14th International Conference, NSS 2020, Melbourne, VIC, Australia, November 25–27, 2020, Proceedings, Springer-Verlag, Berlin, Heidelberg, 2020, p. 112–131. doi:10.1007/978-3-030-65745-1_7.

[4] B. Edelman, Large-scale registration of domains with typographical errors, https://cyber.harvard.edu/archived_content/people/edelman/typo-domains/, 2003. Harvard University.

[5] A. Banerjee, M. S. Rahman, M. Faloutsos, Sut: Quantifying and mitigating url typosquatting, Computer Networks 55 (2011) 3001–3014. doi:10.1016/j.comnet.2011.06.005.

[6] T. Moore, B. Edelman, Measuring the Perpetrators and Funders of Typosquatting, Springer Berlin Heidelberg, 2010, pp. 175–191. doi:10.1007/978-3-642-14577-3_15.

[7] J. Szurdi, B. Kocso, G. Cseh, J. Spring, M. Felegyhazi, C. Kanich, The long "Taile" of typosquatting domain names, in: 23rd USENIX Security Symposium (USENIX Security 14), USENIX Association, San Diego, CA, 2014, pp. 191–206. URL: https://www.usenix.org/conference/usenixsecurity14/technical-sessions/presentation/szurdi.

[8] Y.-M. Wang, D. Beck, J. Wang, C. Verbowski, B. Daniels, Strider Typo-Patrol: Discovery and analysis of systematic Typo-Squatting, in: 2nd Workshop on Steps to Reducing Unwanted Traffic on the Internet (SRUTI 06), USENIX Association, San Jose, CA, 2006, p. 6. URL: https://www.usenix.org/conference/sruti-06/strider-typo-patrol-discovery-and-analysis-systematic-typo-squatting.

[9] P. Agten, W. Joosen, F. Piessens, N. Nikiforakis, Seven months' worth of mistakes: A longitudinal study of typosquatting abuse, in: Proceedings 2015 Network and Distributed System Security Symposium, NDSS 2015, Internet Society, 2015, p. 13. doi:10.14722/ndss.2015.23058.

[10] A. Banerjee, D. Barman, M. Faloutsos, L. N. Bhuyan, Cyber-fraud is one typo away, in: IEEE INFOCOM 2008 - The 27th Conference on Computer Communications, 2008, pp. 1939–1947. doi:10.1109/INFOCOM.2008.258.

[11] V. I. Levenshtein, Binary codes capable of correcting deletions, insertions, and reversals, Soviet physics. Doklady 10 (1965) 707–710. URL: https://api.semanticscholar.org/CorpusID:60827152.

[12] F. J. Damerau, A technique for computer detection and correction of spelling errors, Commun. ACM 7 (1964) 171–176. doi:10.1145/363958.363994.

[13] G. Navarro, A guided tour to approximate string matching, ACM Comput. Surv. 33 (2001) 31–88. doi:10.1145/375360.375365.

[14] T. Ronda, S. Saroiu, A. Wolman, Itrustpage: a user-assisted anti-phishing tool, SIGOPS Oper. Syst. Rev. 42 (2008) 261–272. doi:10.1145/1357010.1352620.

[15] I. Ahmad, M. A. Parvez, A. Iqbal, Typowriter: A tool to prevent typosquatting, in: 2019 IEEE 43rd Annual Computer Software and Applications Conference (COMPSAC), volume 1, 2019, pp. 423–432. doi:10.1109/COMPSAC.2019.00068.

[16] M. Stampar, Blackbook: a historical (black)list of malicious domains, https://github.com/stamparm/blackbook, 2024.

[17] DataForSEO, Top 1000 websites by ranking keywords, https://dataforseo.com/free-seo-stats/top-1000-websites, 2024.

[18] F. Breitinger, B. Guttman, M. McCarrin, V. Roussev, D. White, Approximate matching: definition and terminology, National Institute of Standards and Technology, 2014. doi:10.6028/nist.sp.800-168.

[19] AIL project, Ail-typo-squatting, https://github.com/typosquatter/ail-typo-squatting, 2023.

[20] R. Dhamija, J. D. Tygar, The battle against phishing: Dynamic security skins, in: L. F. Cranor (Ed.), Proceedings of the 1st Symposium on Usable Privacy and Security, SOUPS 2005, Pittsburgh, Pennsylvania, USA, July 6-8, 2005, volume 93 of *ACM International Conference Proceeding Series*, ACM, 2005, pp. 77–88. doi:10.1145/1073001.1073009.