Tracing and Analyzing Web Access Paths Based on User-Side Data Collection: How Do Users Reach Malicious URLs?

Takeshi Takahashi[†], Christopher Kruegel[‡], Giovanni Vigna[‡], Katsunari Yoshioka[*], Daisuke Inoue[†]
[†]*National Institute of Information and Communications Technology,*
takeshi_takahashi@ieee.org, dai@nict.go.jp
[‡]*University of California, Santa Barbara, {chris, vigna}@cs.ucsb.edu*
[*]*Yokohama National University, yoshioka@ynu.ac.jp*

Abstract

Web access exposes users to various attacks, such as malware infections and social engineering attacks. Despite ongoing efforts by security and browser vendors to protect users, some users continue to access malicious URLs. To provide better protection, we need to know how users reach such URLs. In this work, we collect web access records of users from their using our browser extension. Differing from data collection on the network, user-side data collection enables us to discern users and web browser tabs, facilitating efficient data analysis. Then, we propose a scheme to extract an entire web access path to a malicious URL, called a hazardous path, from the access records. With all the hazardous paths extracted from the access records, we analyze web access activities of users considering initial accesses on the hazardous paths, risk levels of bookmarked URLs, time required to reach malicious URLs, and the number of concurrently active browser tabs when reaching such URLs. In addition, we propose a preemptive domain filtering scheme, which identifies domains leading to malicious URLs, called hazardous domains. We demonstrate the effectiveness of the scheme by identifying hazardous domains that are not included in blacklists.

1 Introduction

The Internet has become indispensable social infrastructure, and many people access the web multiple times a day in the course of their daily lives. However, web access exposes users to diverse types of attacks, including malware infections and social engineering attacks. Various techniques have been investigated and implemented to minimize the risk of accessing malicious URLs. In particular, browsers have become more resistant to malware infections and other web-based attacks. For example, the Google Chrome browser uses Google's blacklist service, i.e., Google Safe Browsing (GSB) [1], to minimize users' access to malicious URLs. However, users still reach malicious URLs through web browsing.

To better understand how users reach malicious URLs and to improve user protection, in this paper, we introduce a per-user data analysis scheme and demonstrate its usability. First, we collect access records of each user using our browser extension. User-side data collection enables us to access data that network-side data collection cannot access, i.e., user IDs, browser tab IDs, and navigation information, which, in turn, enable us to efficiently analyze the data in detail. Then, we propose a scheme to extract an entire web access path to a malicious URL, called a hazardous path. The proposed scheme takes advantage of collected data that identify users and browser tabs to efficiently reconstruct the hazardous paths. The first accesses of the paths, i.e., entry points, are identified based on the navigation information that identifies the cause of the navigation. With the extracted access path information, we analyze the web access activities of users who reach malicious URLs, i.e., victims[1], to better understand them. First, we analyze the entry points of hazardous paths to understand the proportion of accesses via bookmarks. We then analyze the risk level of bookmarked URLs. We also analyze the time required to reach malicious URLs and the number of concurrently active browser tabs to demonstrate the usability of data collected at the user side for further analysis.

In addition, we propose a preemptive domain filtering scheme. This scheme determines the risk level of accessing each domain by analyzing the hazardous paths and identifies domains that lead to malicious URLs, i.e., hazardous domains. These domains are not blacklisted; however, the proposed scheme suggests filtering traffic on them because, even if they do not host any malicious contents themselves, the access paths thereafter lead to malicious URLs.

Contributions The primary contributions of this work are summarized as follows.

1. We describe a user-side data collection approach using a browser extension. Differing from earlier work that

[1] These users are not necessarily harmed by accessing malicious URLs; however, for convenience, we use the term "victim" for such users.

collects traffic data on network devices, user-side data collection enables us to access to a broader range of data, including user IDs, browser tab IDs, and user navigation information, allowing us to efficiently analyze user behaviors.

2. We introduce a scheme that reconstructs hazardous paths from the collected data. The scheme repeatedly traces previous accesses until it reaches the entry points of hazardous paths to reconstruct the paths. The entry points are determined by looking up user navigation information. Differing from earlier work that does not discern users and browser tabs, this scheme minimizes ambiguity by discerning them, narrowing down the lines of logs that need to be analyzed.
3. We analyze users' browsing behaviors to demonstrate the usability of data obtained from browsers, i.e., tab IDs and user navigation information. The analysis reveals that bookmark access is the major type of initial access on hazardous paths. In the collected data, bookmark access occupies the largest share of entry point types on hazardous paths, and its share is greater on hazardous paths than the share on all paths, including hazardous and non-hazardous paths.
4. We analyze the risk level of bookmarked URLs by defining a parameter that indicates the certainty of reaching malicious URLs, We show that there are bookmark entries that surely lead to malicious URLs, which indicates that reviewing and sanitizing bookmark entries may minimize the risk of reaching malicious URLs.
5. We introduce a preemptive domain filtering scheme that filters traffic before users reach malicious URLs. Our analysis shows that there are non-malicious domains that often lead users to malicious URLs. To identify such domains, the scheme calculates the risk levels of all domains appeared on hazardous paths and identifies domains that are likely to navigate users to malicious URLs. The scheme may filter traffic traversing on these domains, or at least provide alerts to protect users from reaching malicious URLs.

To the best of our knowledge, this is the first academic paper that analyzes users' web access records by discerning users and browser tabs and by considering user navigation information. In particular, we focus on the entry points of hazardous paths, revealing the importance of reviewing bookmarks. The proposed preemptive domain filtering scheme is also unique, which identifies hazardous domains that are not included in blacklists.

Organization of this paper The remainder of this paper is organized as follows. Section 2 presents our data collection scheme. Section 3 introduces a scheme to reconstruct an entire hazardous path by iteratively tracing previous accesses until reaching path entry points, and Section 4 presents an analysis of users who reach malicious URLs, A scheme to preemptively filter traffic based on domain risk levels is introduced in Section 5. Section 6 considers issues that are not addressed in the earlier sections. Related work is reviewed in Section 7, and Section 8 concludes this paper.

2 User-Side Data Collection

Web access records are collected at the user side using a browser extension developed in-house [2]. The browser extension runs on the Chrome browser, and anyone who agrees to the terms and conditions can install and use the browser extension. This browser extension works as a sensor that records each user's web access activities and periodically shares the recorded information to a server. The browser extension is in Japanese; therefore, we assume that the users are primarily Japanese speakers. In this section, the list of the collected information, the ethical considerations, the access log complementation, and the dataset generated for analysis are elaborated.

2.1 Collected Data

Our browser extension uses Google Chrome's APIs [3] to collect web access data of a user from the web browser. The browser extension uses the chrome.webRequest API [4] to observe and analyze traffic and the chrome.webNavigation API [5] to receive notifications about the status of navigation requests. The following data are collected.

1. **URL**: the URL of the requested document obtained from the HTTP request header
2. **Timestamp**: the UNIX time when the request is issued
3. **Referer**: the referer value obtained from the HTTP request header
4. **Tab ID**: the identifier of a browser tab. Its uniqueness is only guaranteed during the session.
5. **Tab URL**: URL shown on the browser tab
6. **Resource type**: the type of resource defined by the chrome.webRequest API. It takes one of the following values: "csp_report," "font," "image," "main_frame," "media," "object," "ping," "script," "stylesheet," "sub_frame," "websocket," or "xmlhttprequest." Note that main frame, which is identified by the value "main_frame," is a document that is loaded for a top-level frame.
7. **Transition type**: the cause of the navigation defined by the chrome.history API [6]. It takes one of the following values: "auto_bookmark," "auto_subframe," "form_submit," "generated," "keyword," "keyword_generated," "link," "manual_subframe," "reload," "start_page," or "typed."

In addition to the data listed above, the browser extension and server provide the following supplementary information.

1. **User ID**: unique user identifier
2. **Source tab ID**: the identifier of the tab that has generated the current tab. It is recorded when a new window is generated, for instance, by target="_blank" and window.open().
3. **GSB evaluation results**: an indicator whether the URL is listed by GSB. The server aggregates all URLs collected in a single day and queries GSB regarding their maliciousness once a day.
4. **Alexa Traffic Rank**: traffic rankings provided by Alexa Top Site [7]. The server aggregates all URLs collected in a single day and queries Alexa Top Site regarding their ranking. Note that we check up to the top 1,000 sites.

One access record consists of the above 11 items. Note that our browser extension can collect a broader range of data; however, data not relevant to this work have been omitted.

2.2 Ethical Considerations

We worked with our Internal Review Board to ensure that our usage of the logs was ethical and respectful of users' privacy. We defined the terms and conditions for our browser extension [2]. All collected data are listed, and we stipulate that we analyze the data collected from the browser extensions to detect and prevent access to malicious URLs. Users installing the browser extension need to agree to the terms and conditions.

The collected data contains privacy-related details; therefore, we have strict restrictions on its use. Any personally identifiable user information was expunged or coded before records were stored on the servers. The user ID recorded in the log is an internal number unique to each user and cannot be directly linked to any personally identifiable information. Raw URLs cannot be shared with external parties. Therefore, we do not use VirusTotal [8], which requires us to submit raw URLs. Instead, we use GSB to evaluate the maliciousness of URLs because it does not require us to upload raw URLs. In addition, we delete all records of users who request that their records be deleted.

The logs we use in our analysis are stored on a server in a secure facility, and only registered users from registered machines that implement adequate security measures can access them. No raw data are allowed to be copied outside the machines; therefore, all analyses must be conducted on the secure servers. Only aggregated results were exported from the secure server for further analysis.

2.3 Access Log Complementation

Our scheme collects users' web access data on a granular scale, which enables us to analyze their behavior in detail. However, we sometimes fail to collect certain data. We have identified three factors influencing these failures. First, the implementation of the data collection module was imperfect. A user may take an explicit action to move to another page before the browser receives a response to a request. In this case, some fields of the access record are left blank, or the entire record is not recorded. Second, users can specifically avoid data collection for certain cases. Users can specify the list of URLs and domains that they do not want our browser extensions to record. The browser extension also does not record logs if a user accesses the web in incognito mode. These functionalities are provided to protect users' privacy. In addition, some users may deactivate the browser extension when accessing sites that they do not want us to monitor. Third, the Chrome browser does not provide the intended records when its processing burden becomes high.

Our scheme complements missing main frame entries in the log because these entries play a core role in our analysis. Typically, access to a tab URL produces a series of requests for multiple content, e.g., main frame, subframe, image, script, and style file. However, the log sometimes lacks main frame that should appear. In this case, we generate a complementary main frame entry, where its tab URL and URL are both set to the tab URL. Here the timestamp is set to that of the first entry in the consecutive access records for the tab URL.

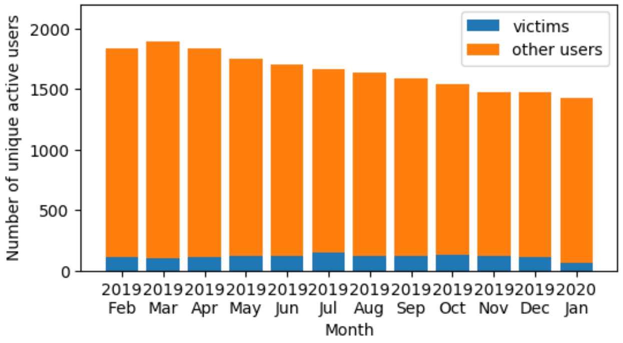

Figure 1: Number of unique active users per month

2.4 Dataset

The dataset we used consists of data collected from February 1, 2019 to January 31, 2020 (12 months). During the experimental period, 4,306,529,287 access records were collected, among which were 76,474 accesses to malicious URLs. Figure 1 shows the number of unique active users per month. Note, a user is considered active if any browsing activity of the user is logged. On average, 831 users accessed the web at least once a day, 1,650 users accessed the web at least once a month, 1,013 users accessed the web more than seven days a month, and 115 users (victims) accessed malicious URLs at least once a month.

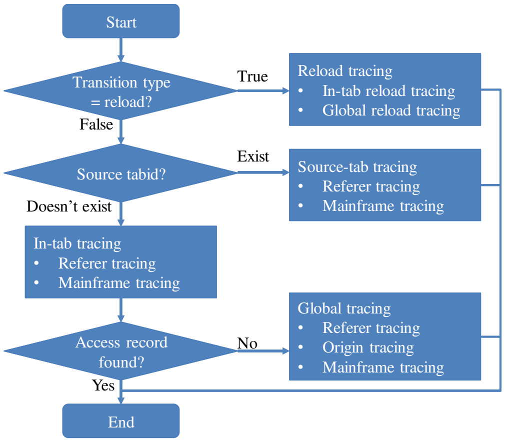

Figure 2: Process flow for tracing previous accesses

3 Access Path Reconstruction

This section introduces our access path reconstruction scheme. It extracts a hazardous path by recursively tracing previous accesses until the entry point to the path is identified. We also present case studies in this section to demonstrate the effectiveness of the scheme.

3.1 Previous Access Tracing

Our access path reconstruction scheme traces previous accesses by using user IDs and tab IDs provided by our dataset to minimize the scope of analysis. Figure 2 shows an overview of the tracing process. If the transition type of an access record is set to "reload," the access is considered as a page reload, and the reload tracing method is run to identify the access record of the reloaded page. Otherwise, if source tab ID is set, this access is considered to be originated from another tab, and the source-tab tracing method is run to identify the record of previous access on the source tab. Otherwise, the in-tab tracing method is run to identify the previous access record within the current tab. If no proper record is found, the global tracing method is run to identify the previous access record from the user's entire past access records. In most cases, our scheme analyzes a single user's access records in one browser tab, so the analysis is more efficient than the one that do not minimize the scope of the analysis. The details of the four methods mentioned above are elaborated below.

Reload tracing A "reload" transition type indicates that the page is reloaded. This includes the following cases: (1) The reload button/menu on the browser was pressed/selected, (2) the same URL as the previous access was entered in the address bar of the browser, (3) the session was reconstructed by selecting one of the recently closed tabs in the browser's history menu, and (4) a browser set to continue where its user left off on startup was started. In all cases, the tab URL and URL of the previous access record remain the same as these of the current access record. For the first two cases, the tab ID remains the same; however, the latter two cases do not guarantee that the tab ID remains the same. If the transition type of an access record is set to "reload," our access path reconstruction scheme runs the reload tracing method, which determines the latest past access record with the same tab URL and URL values as the current access record. It first retrieves the past access records of the user within the same browser tab, and then retrieves all past access records of the user if no matching entry is found.

Source-tab tracing When an access record provides a source tab ID, our access path reconstruction scheme runs the source-tab tracing method, which analyzes the past access records of a user on the source tab to identify the previous access record. When the access record provides a referer information, the method traces the records between the current access and the latest past main frame access and determines the latest past access record whose URL matches the referer as the previous access record (referer tracing). If referer information is unavailable or no suitable record is found, the method determines the latest past main frame access record as the previous access record (main frame tracing).

In-tab tracing When an access record does not provide a source tab ID, our access path reconstruction scheme runs the in-tab tracing method, which analyzes the past access records of a user on the current tab to identify the previous access record. As with source-tab tracing method, the in-tab tracing method analyzes the records by running referer tracing and main frame tracing techniques to identify the previous access record. Note that most previous access records will be identified by this method because new tab creation is less frequent than access on the current tab.

Global tracing When no previous access was found by the in-tab tracing method, our access path reconstruction scheme runs the global tracing method, which analyzes the past access records of a user on all tabs to identify the previous access record. As with the source-tab and in-tab tracing methods, the global tracing method first runs referer tracing if referer information is available. There are cases the referer contains only its origin information. The global tracing method uses it if no suitable entry is found by referer tracing. The method determines the latest past main frame access record whose URL value includes the referer, i.e., origin, as the previous access (origin tracing). If referer information is unavailable or no suitable record is found, the method may run main frame tracing; however this does not necessarily provide a reliable trace and is not used in the analysis of this paper. Note that the origin tracing is not used by the source-tab and in-tab tracing methods because they can identify a reliable previous access record by running the main frame tracing technique.

3.2 Identifying access path entry points

The processes described in Section 3.1 are iterated until they identify the first access of the hazardous path, i.e., the entry point. We regard an access that do not follow links and is discontinuous from the previous access as an entry point. The following types of accesses are considered as entry points.

Bookmark access A user may jump to the desired page by selecting a bookmark entry on the browser. We determine that the user came to the page by selecting a bookmark when the transition type is set to "auto_bookmark."

Session reconstruction Sessions can be reconstructed, for example, by selecting one of the recently closed tabs in the browser's history menu. We determine that the user came to the page by reconstructing a past session if the transition type was set to "reload" and no access was found on the same tab for more than a predefined amount of time. Note that we do not include the access records prior to session reconstruction in the hazardous path in this paper, but we could include such access records in the path for different analyses.

Web search A user can find various pages of interests by submitting a new search query on general web search engines. We determine that a new web search is initiated if (1) the URL is one of the top pages of major search engines[2], or (2) the transition type is "form_submit" and the URL is a search result page of major search engines[3]. Access records with a source tab ID are excluded because there must be precedent accesses in the specified tab. Note that site-specific search engines that are often available for many streaming services and pornography sites are not included.

Omnibar access The omnibar on the Chrome browser combines an address bar with the Google search box. Users can use the omnibar to initiate a web search or access their browsing history. The omnibar also suggests keywords to improve search results. The transition type "generated" identifies access that is based on the selection of choices provided by the omnibar.

Address typing Users often enter a new URL to initiate another browsing activity; however, as part of ongoing browsing activity, they also often modify the current URL in the address bar to browse another page, e.g., the top page of the current site. Therefore, we consider the page with the "typed" transition type as an entry point if the domain of the page differs from the previous page. Note that if the typed URL is one of the major search engines, we see it as an initiation of a new access path; however, we label the access as a web search rather than new URL typing.

Start page access When the Chrome browser starts, the page specified by the program argument or set as the default opens. If the transition type is "start_page," we consider that the browser was launched and that the page was specified in the program argument or set as the default page. Note that the transition type is set to "start_page" if a user accesses a link on an external application, e.g., an email application, because the Chrome browser is then launched with the URL of the link as its argument on the OS.

3.3 Case Studies

We extract hazardous paths by applying the techniques described in Sections 3.1 and 3.2. Two cases are described in this section to demonstrate their effectiveness.

Table 1 shows an entire hazardous path that reaches a URL labeled "SOCIAL_ENGINEERING" by GSB. First, the in-tab tracing method identified access to a subframe page in the same tab at 16:14:40 as its previous access using referer information. It then identified access to a main frame page in the same tab at 16:14:39 as its previous access using referer information. Then, the global tracing method identified access to a main frame page at 16:14:13 in another tab as its previous access using referer information. The in-tab tracing method then identified access to a main frame page at 16:13:54 in the same tab as its previous access using referer information. Now, we see that the page's transition type was set to "auto_bookmark;" thus we consider this page as the initial access of the hazardous path.

Table 2 shows an entire hazardous path that reaches a URL labeled "MALWARE" by GSB. Our in-tab tracing method identified the latest past main frame access in the same tab at 15:33:37 as its previous access. Note that the record of the main frame access was complemented using the scheme mentioned in Section 2.3. Our in-tab tracing method then identified the latest past main frame accesses in the same tab at 15:33:26, 15:32:35, and 15:32:08 as the previous accesses. The source-tab tracing method then identified the latest past main frame access on the source tab at 15:31:50 as its previous access. Similarly, the in-tab tracing method identified previous accesses by monitoring main frame and referer information. Identification of these previous accesses led to the initial access, which was accessed from a bookmark, at 15:26:57.

[2] In this work, we use the following URLs to determine the top pages of major search engines: https://www.baidu.com/, https://www.bing.com/, https://duckduckgo.com/, https://www.google.co.jp/, https://www.google.com/, https://www.yahoo.co.jp/, and https://www.yahoo.com/.

[3] In this work, we determine URLs containing one of the following strings as search result pages: https://www.baidu.com/s?, https://www.bing.com/search, https://duckduckgo.com/?q=, https://www.google.co.jp/, https://www.google.com/search?, https://search.yahoo.co.jp/search, or https://search.yahoo.com/search.

Table 1: Hazardous path on August 19, 2019 ("SOCIAL_ENGINEERING" label was set by GSB)

Time (JST)	Tab ID	URL(cropped)	Source tab ID	Transition type	Resource type	Tracing method
16:13:54	193	https://avgle.com/video/UNXIII	—	auto_bookmark	mframe	in-tab (referer)
16:14:13	193	https://avgle.com/search/video	—	form_submit	mframe	global (referer)
16:14:39	200	https://avgle.com/video/odaqWq	—	link	mframe	in-tab (referer)
16:14:40	200	https://olmsoneenh.info/ajWpZ.	—	auto_subframe	sub_frame	in-tab (referer)
16:14:41	200	https://10-81.s.cdn15.com/cr/3	—	—	image	—

mframe: main_frame

Table 2: Hazardous path extracted on August 28, 2019 ("MALWARE" label was set by GSB)

Time (JST)	Tab ID	URL(cropped)	Source tab ID	Transition type	Resource type	Tracing method
15:26:57	182	http://javtorrent.re/category/		auto_bookmark	mframe	in-tab (referer)
15:27:14	182	http://javtorrent.re/?s=080819		form_submit	mframe	in-tab (referer)
15:27:26	182	http://javtorrent.re/uncensore		link	mframe	in-tab (mframe)
15:28:28	182	http://javtorrent.re/?s=HEYZO-		form_submit	mframe	in-tab (mframe)
(omitted 18 access records)						
15:31:50	182	http://javtorrent.re/uncensore		link	mframe	source-tab (mframe)
15:32:08	403	https://www.google.com/search?	182	link	mframe	in-tab (mframe)
15:32:35	403	https://7mmtv.tv/zh/uncensored		link	mframe	in-tab (mframe)
15:33:26	403	https://www.google.com/search?		link	mframe	in-tab (mframe)
15:33:37	403	http://javhuge.com/Momoki%20		—	complemented	in-tab (mframe)
15:33:37	403	http://javhuge.com/zb_users/th		—	stylesheet	—

mframe: main_frame, o_referer: origin-only referer

4 Unveiling User Behavior

Web access data collected at the user side reveals user IDs and browser tab IDs, enabling us to efficiently reconstruct access paths, as described in Section 3. It also provides data that are not collected on the network, such as transition type. With the transition type information, we can identify access path entry points as described in Section 3.2, and such information can be used to analyze user behavior. In this section, we demonstrate the usability of the data collected on the user side by analyzing them and unveiling user behavior. Specifically, we answer the following questions: (1) what are the initial accesses of hazardous paths? (2) what is the risk level of bookmarked URLs? (3) how long does it take for users to reach malicious URLs? and (4) how many active browser tabs do users open when accessing malicious URLs? Answers to these questions may not directly provide any solution for improving user protections; however, they will deepen understanding of victims and will be a basis for future studies.

Table 3: Types of path entries (February 2019–January 2020)

Types of initial accesses	Entries of hazardous paths	Entries of all paths
Bookmark access	3,213 (48.7%)	1,331,170 (38.6%)
Web search	955 (14.5%)	541,046 (15.7%)
Session reconstruction	814 (12.3%)	600,366 (17.4%)
Omnibar access	689 (10.4%)	581,223 (16.8%)
Address typing	646 (9.8%)	89,966 (2.6%)
Start page access	179 (2.7%)	307,536 (8.9%)
Link access	107 (1.6%)	—
Total	6,496 (100%)	4,403,471 (100%)

4.1 Path entry point analysis

Considering the measures implemented by recent browsers, accidentally reaching a malicious URL from the ISP's portal site or via a major search engine has become increasingly infrequent; however, users still reach malicious URLs through web browsing. To understand their browsing activities, we analyze the types of initial accesses on hazardous paths, i.e., hazardous path entry points. Table 3 shows the breakdown of the types of initial accesses on hazardous paths over 12 months. These types are identified based on the definition provided in Section 3.2. For reference, it also shows the breakdown of the types of initial accesses on all paths including hazardous and non-hazardous paths.

"Bookmark access" was the most frequent entry point. Many users reach malicious URLs via a bookmark, which shortens the path from a portal site or search engine to malicious sites. "Web search" was the second most frequent entry point. Users enter a hazardous path by submitting a query to a general search engine. One typical scenario of this entry type was that users obtained a keyword from the web before they submitted a query to a search engine. For example, while browsing the adult section of a legitimate shopping site, a user found the identifier of a pornographic video, e.g., product code. The identifier was used to retrieve the web and, consequently, the user reached an illegitimate site, which eventually lead to a malicious URL. "Session reconstruction" accounted for third most frequent entry points. If we combine the paths before and after the session reconstructions, the path will become lengthy. Naturally, a lengthy path has a higher probability of reaching malicious URLs because more sites are visited, including those that are many hops away from the search engine result pages or browsers' start pages. "Omnibar access" was the fourth most frequent entry point, followed by "address typing." All of these path entry types shorten the path between the portal site or search engine and the malicious site. In this context, they are shortcuts in the access path to a malicious URL. "Start page access" was also found to be an entry point to hazardous paths; however, the number of this type of entry point is small compared with the other types.

Apart from these, several entry points are labeled as "link access." These are cases where the transition type of the access record is "link," though no appropriate previous access is found within the log we analyzed, making the access record discontinuous. Considering the meaning of the label "link" assigned by the Google API, there should be some previous accesses prior to the entry. The lack of previous accesses may occur due to either of the following reasons. First, previous access was performed prior to the first log record. Accesses prior to the first day of the month could not be traced because we conduct per-month analysis in this work. Second, the collected logs are incomplete, be it intentionally or accidentally, as we discussed in Section 2.3.

Figure 3 shows the breakdown of the types of initial accesses on hazardous paths for each month. The general trend of the breakdown is the same during these months except July 2019. There were several users who typed many URLs in the address bar of the browser in July 2019 and reached malicious URLs, though their motivations were unknown.

4.2 Bookmarked URL analysis

Table 4 shows the domains of bookmarked URLs, i.e., bookmark domains, that most frequently reach malicious URLs in descending order of the number of accesses. Note that only the accesses through the selection of bookmark entries are counted. As can be seen, the list includes pornography sites, illegal book/manga sharing sites, and file sharing sites. However, it also contains legitimate search engines, such as google.com and yahoo.co.jp. Considering the total number of accesses on any path, the percentage of accesses that reach malicious URLs remains very small; thus the risk level is small. Therefore, the risk level of domains cannot be evaluated by the number of accesses that reach malicious URLs.

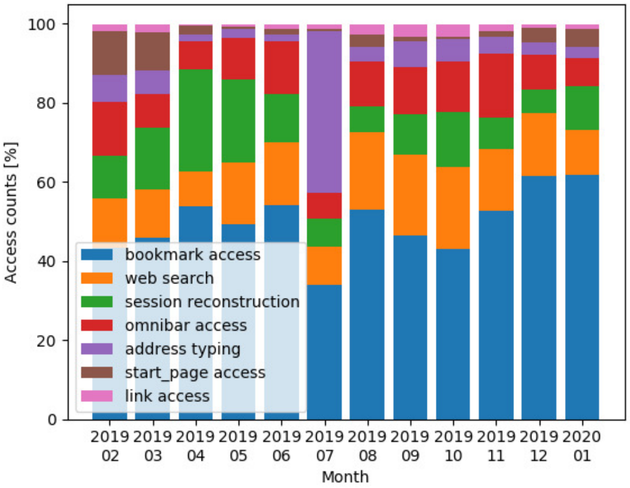

Figure 3: Breakdown of the types of path entries

Table 4: Top 10 bookmark domains that most frequently reach malicious URLs (February 2019–January 2020)

Bookmarked	Access counts on		Risk
domains	hazardous paths	all paths	level
avgle.com	2,184	3,566	61.25
xbooks.to	136	1,260	10.79
google.co.jp	113	69,878	0.16
yahoo.co.jp	89	191,297	0.05
bejav.net	79	79	100.00
javmost.com	37	37	100.00
mac-torrent-download.net	34	68	50.00
13dl.net	33	3,389	0.97
smv.to	19	251	7.57
youtube.com	18	95,572	0.02

To identify untrustworthy bookmark domains, we define the risk level of a domain that shows the certainty of reaching malicious URLs as follows:

$$R(domain) = \frac{NbrAccess_{malurl}(domain)}{NbrAccess_{all}(domain)} \tag{1}$$

where $R(domain)$ represents the risk level of a domain, $NbrAccess_{malurl}(domain)$ is the number of accesses to the domain that eventually reach malicious URLs, and $NbrAccess_{all}(domain)$ is the number of all accesses to the domain. Note that these parameters ignore whether a malicious URL is in the next hop or in multiple hops away. With this risk level, we evaluate the risk level of bookmark domains.

Table 5 shows the bookmark domains that most frequently

Table 5: Top 10 untrustworthy bookmark domains (February 2019–January 2020)

Bookmarked	Access counts on		Risk
domains	hazardous paths	all paths	level
bejav.net	79	79	100.00
javmost.com	37	37	100.00
xdytt.com	8	8	100.00
91mjw.com	6	6	100.00
incestflix.com	6	6	100.00
theyoump3.com	6	6	100.00
gofucker.com	5	5	100.00
anipo.tv	3	3	100.00
javbraze.com	3	3	100.00
avdvd.tv	2	2	100.00

Table 6: Top 10 Bookmark domains (February 2019–January 2020)

Bookmarked	Access counts on		Risk
domains	hazardous paths	all paths	level
yahoo.co.jp	89	191,297	0.05
youtube.com	18	95,572	0.02
google.co.jp	113	69,878	0.16
amazon.co.jp	8	45,326	0.02
google.com	6	33,494	0.02
twitter.com	3	29,159	0.01
facebook.com	2	27,387	0.01
nicovideo.jp	5	25,216	0.02
livedoor.jp	1	20,829	0.00
rakuten.co.jp	1	17,692	0.01

reach malicious URLs between February 2019 and January 2020 in descending order of the risk level. As can be seen, legitimate sites that appeared in Table 4 do not appear in the table. The top 10 bookmark domains accessed between February 2019 and January 2020 are listed in Table 6. As can be seen, the risk levels of the domains in Table 5 are significantly higher than the risk levels in Table 6.

Based on these analyses, we could review bookmarks to minimize the risk of users. As discussed in Section 4.1, bookmark access is a major entry point to hazardous paths. Therefore, we can expect to minimize the number of accesses to malicious URLs by sanitizing bookmarks. It can be difficult to prevent users from adding hazardous URLs to bookmarks or to block access to hazardous bookmarked URLs. However, simply showing alerts when accessing hazardous bookmarks may help to improve the situation. Regularly reviewing the list of bookmarks and providing alerts will be effective.

4.3 Time to reach malicious URLs

Figure 4 shows the cumulative histogram of the time to reach a malicious URL from users' first access of the paths. Among all the hazardous paths between February 2019 and January 2020, 87.03% reached a malicious URL within 24 hours from their initial accesses, 80.03% within 6 hours, 68.75% within an hour, and 60.31% within half an hour. We conjecture that many accesses to malicious URLs occur within an hour, or even within half an hour, because victims already have the URLs that are close to malicious URLs in their bookmarks, or they already know hazardous keywords that may lead to malicious URLs when initiating a web search, as seen in Section 4.1.

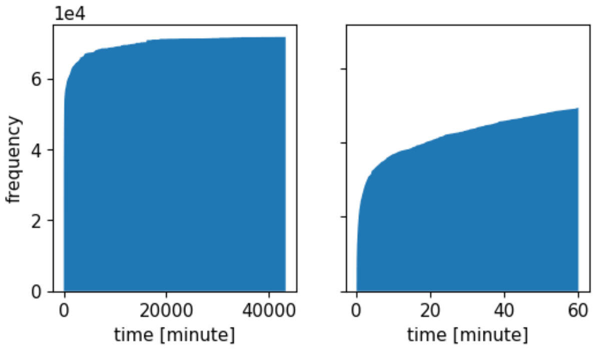

Figure 4: Normalized cumulative histograms of the time to reach a malicious URL in February 2019–January 2020 (left: complete histogram, right: first hour excerpt of the histogram)

4.4 Number of active browser tabs

This section analyzes the number of active browser tabs. A browser tab is considered active if some activities of a user is observed on the browser tab. We assume that the number of active browser tabs does not change before or after accessing ordinary URLs. However, it is observed that many malicious sites and some other sites open browser tabs unnecessarily. To confirm that, we analyzed the number of active browser tabs before and after malicious URL accesses.

Figure 5 shows a histogram of the number of active browser tabs within 10 minutes before and after malicious URL access. Over 99% of victims use 1–2 browser tabs before and after visiting malicious URLs. The figure also shows the number of active browser tabs within 10 minutes before a malicious access. Over 91% of victims use 1–4 browser tabs within 10 minutes before accessing malicious URLs[4].

The difference in the number of active browser tabs between the above two histograms indicates that there could be browser tabs generated to reach malicious URLs, e.g., redirection pages opened in a new browser tab. This difference cannot be used as a feature for detecting malicious URL access because it requires access records after the malicious URL access, but it could be analyzed further to devise a

[4] Note that the maximum number of active browser tabs was 39; however, the frequency was negligible. Thus active browser tabs greater than 18 are not present in the histogram.

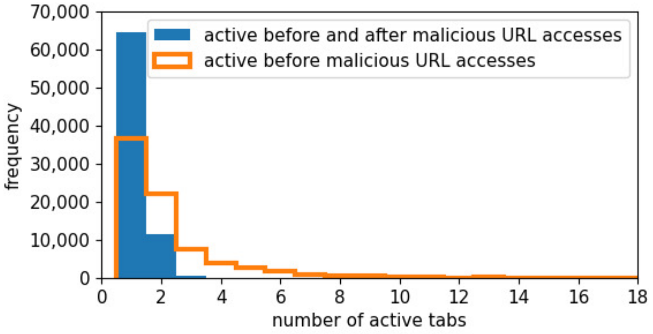

Figure 5: Histogram of the number of active browser tabs (February 2019–January 2020): browser tabs active within 10 minutes before and after malicious URL accesses and those before malicious URL accesses

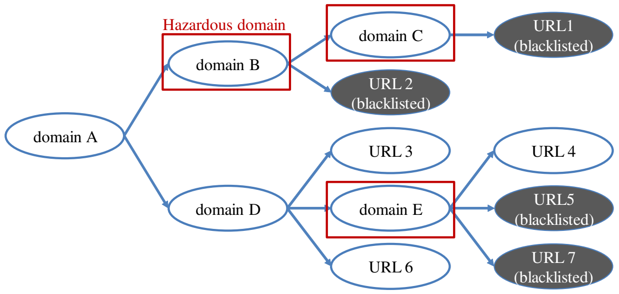

Figure 6: Concept of preemptive domain filtering scheme

feature to detect malicious URL access.

5 Preemptive Domain Filtering Scheme

The hazardous paths generated by the scheme proposed in Section 3 could be used to improve user protection. In this section, we propose a preemptive domain filtering scheme based on the risk level evaluation of domains on the hazardous paths.

Malicious URLs are included in blacklists; however, there are non-blacklisted URLs that lead to malicious URLs. We refer to domains that have high probability of leading to malicious URLs as hazardous domains. Access to malicious URLs can be minimized by identifying such domains and taking countermeasures, e.g., adding the domains to blacklists or alerting users. Figure 6 shows the concept of the proposed preemptive domain filtering scheme. In this figure, an access path tree consisting of multiple access paths that reach seven different URLs is described, and four of the URLs are included in a blacklist. The proposed scheme calculates the probability of reaching malicious URLs from a domain and identifies the domain as hazardous if the probability is greater than a certain threshold value. Accesses through domains B and C reach only blacklisted URLs; thus they are regarded as hazardous domains. Accesses through domain E reach both blacklisted and non-blacklisted URLs; however this domain is regarded as hazardous because the probability is greater than the threshold value. Other domains are regarded as non-hazardous because the probability is less than the threshold value. These hazardous domains are not included in the blacklist; however, we could block access to these hazardous domains or alert users to minimize the risk of users' accessing malicious URLs.

Table 7: Top 10 newly identified untrustworthy domains with a risk level above 50% (February 2019–January 2020)

Bookmarked domains	Access counts on hazardous paths	Access counts on all paths	Max risk level	Number of months
avgle.com	26,971	80,766	50.27	12
xbooks.to	1,185	11,138	66.66	4
codeday.me	991	1,105	100.00	11
ero-advertising.com	323	1,950	52.72	6
aphookkensidah.pro	272	1,043	100.00	12
tubepornclassic.com	109	782	77.14	5
highporn.net	105	152	69.07	1
erodoujin-index.net	82	109	75.22	1
fbk.tokyo	39	228	66.66	10
avli.me	38	58	65.51	1

Table 8: Top 10 newly identified untrustworthy domains with a risk level of 100% (February 2019–January 2020)

Bookmarked domains	Access counts on hazardous paths	Access counts on all paths	Number of months
codeday.me	991	1,105	11
aphookkensidah.pro	272	1,043	12
collectionanalyser.com	32	65	3
vidia.tv	27	33	3
livetotal.tv	27	104	4
jqaaa.com	27	28	3
eimusics.com	25	63	5
dentaint.pro	23	30	5
livetotal.net	22	52	4
javbraze.com	20	20	2

To realize this preemptive domain filtering, we first reconstruct hazardous paths using the proposed scheme described in Section 3. We then extract all domains on the paths. Note that all domains on the path reach more than one malicious URL, while domains that did not appear on hazardous paths are not known to reach any malicious URLs. Finally, we analyze the risk levels of the URLs on hazardous paths using the risk level $R(domain)$ defined in Section 4.2.

5.1 Identifying hazardous domains

We evaluated risk levels of all domains on hazardous paths using the proposed preemptive domain filtering scheme. Table 7 shows the top 10 newly identified domains with a risk level above 50%, and Table 8 shows those with a risk level of 100% over 12 months. Note, domains already identified by GSB have been excluded. The maximum risk level column shows the maximum value of monthly risk levels of a domain during the 12 months, while the number of months column shows the number of months the domain appeared on hazardous paths[5]. As can be seen, the proposed scheme can identify non-blacklisted domains that most likely navigate users to malicious URLs.

Table 9 shows the breakdown of the number of domains on hazardous paths by risk levels. To protect users, if the risk level of a domain is above a certain threshold, we could filter traffic on a domain or issue alerts. If the threshold is set to 80%, 355 domains are identified as hazardous domains in addition to 619 domains that have already been identified by GSB.

5.2 Blocked URLs

By enforcing the proposed preemptive domain filtering scheme and blocking access to hazardous domains, some URLs will become unreachable. To determine the effectiveness of the blocking, we identified URLs that would have been blocked if we have enforced the scheme using the dataset. Note that the scheme needs to know the threshold value to identify hazardous domains that need to be filtered as we have seen in Section 5.1, and we set the value to 80% in this section. URLs that became unreachable fall into one of the following types.

1. **Blacklisted URLs:** The proposed scheme blocks accesses to the hazardous domains that are located on the way to the blacklisted URLs. Therefore, users cannot reach the URLs, though access to these URLs are anyway blocked by the existing blacklists. Note that we used GSB for the blacklist, but other blacklists can be used in place of GSB to select domains for preemptive filtering.
2. **URLs included in other blacklists:** These were not blacklisted by GSB but were known to be malicious by the other blacklists. Here, we used a proprietary blacklist that identifies malicious URLs based on the signatures on URL strings. However, the number of these URLs remains small. We conjecture it is because policies of blacklists differ each other on what is detected as malicious. Indeed, GSB identifies more social-engineering type URLs and less malware-related URLs than the proprietary blacklist we used.
3. **Non-blacklisted URLs whose domains are the same as the blacklisted URLs:** It may take some time for the GSB to register a malicious URL; therefore the URL may not be registered when a user visits it. These URLs will or should be blacklisted. Sometimes, its domain instead of the URL could be registered on the blacklist.
4. **Unreachable URLs:** These URLs are already unreachable at the time of writing this paper. Malicious URLs often disappear after a period of time. Therefore, the likelihood that these URLs are malicious is not too small.
5. **URLs with illegitimate or harmful content:** Many of the blocked URLs that do not fall into any of the above four categories deal with pornography, scanned manga and books, and music and video files. None of these URLs are listed in the Alexa Top 1,000 sites on a global basis [7] during the entire observation period. Although the proprietary blacklist we used did not recognize these URLs as malicious, determination of maliciousness largely depends on the policy of each blacklist, and these pages are not necessarily legitimate even if they are not included in the blacklist. These pages are likely to be irrelevant for the daily lives of most users. Therefore, the impact of those pages becoming unreachable is limited.

These URLs that were made unreachable are either malicious, unreachable, illegitimate, or harmful; thus blocking them would help improve user protection without impairing their legitimate activities.

6 Discussion and Analysis

Each technique has been discussed and evaluated in earlier sections. This section discusses issues related to our data collection and analysis approaches, limitation of our dataset, and directions for further analyses.

6.1 Advantages of user-side data collection

The uniqueness of our overall approach stems from our user-side data collection, which provides various data of browser users that cannot be collected on the network. These data provide two types of advantages: analytical efficiency and access to user-side data.

1. **Analytical efficiency:** The collected data includes user IDs and browser tab IDs. We can narrow down the data we need to look into by filtering with a user ID and a browser tab ID. In this way, the complexity and ambiguity of the data will be minimized, leading to more accurate analysis. As a result, the cost and time required for the analysis are also minimized.
2. **Access to user-side data:** We can obtain data that are unavailable from data collected on the network,

[5] Note that the GSB periodically revises the evaluation results of blacklisted entries.

Table 9: Evaluation of domains on hazardous paths (February 2019–January 2020)

Risk level	Blacklist	Number of domains												
	coverage	2019											2020	Total
		Feb	Mar	Apr	May	Jun	Jul	Aug	Sep	Oct	Nov	Dec	Jan	
100%	full	51	58	60	72	61	289	28	44	50	74	69	47	903
	partial	1	2	3	1	1	3	0	1	1	2	0	1	15
	none	41	42	44	24	42	69	53	24	30	32	38	28	467
[80%-100%)	partial	0	2	1	1	2	2	1	0	0	3	1	2	15
	none	7	1	3	2	2	1	4	1	1	3	3	1	29
[60%-80%)	partial	3	2	1	2	0	6	4	0	1	0	2	1	22
	none	8	10	5	0	5	11	11	7	4	10	9	4	84
[40%-60%)	partial	0	1	1	1	0	6	0	0	0	1	1	1	12
	none	24	35	25	12	20	29	23	12	19	15	18	14	246
[20%-40%)	partial	3	0	2	3	1	3	2	0	1	1	0	1	17
	none	34	53	34	26	20	36	30	27	24	26	33	22	365
Others	partial	6	5	4	1	4	5	1	2	2	0	0	1	31
	none	333	398	218	192	193	250	250	160	192	183	163	173	2,705
Total		511	609	401	337	351	710	407	278	325	350	337	296	4,912

e.g., users' navigation information, which enables us to analyze users in detail. Indeed, our browser extension can collect data that are not listed in Section 2.1, such as the list of installed browser extensions and process information.

This paper demonstrated the usability of our data collection and analysis approach, and we hope this paper will encourage per-user data analysis to better protect users.

6.2 Attracting and Motivating Users

In this work, the primary is the number of users who install our browser extension and continue using it. The browser extension collects user's privacy-related information, which may reveal information they do not want anybody to know. Although the terms and conditions state that we do not link the data and the user's identity, this may discourage people from installing the browser extension.

To motivate users to install our browser extension, we implemented a campaign in the past, where a user can obtain a JPY 2,000 Amazon gift card. The campaign was successful from the standpoint of encouraging users to install the browser extension; however, over half of the users stopped using the browser extension within three months, indicating that the campaign was unable to motivate users to continue using the software.

Rather than asking people to install and use the browser extension, we redesigned the browser extension so that people would be interested in installing and continuing to use it by using a popular character, called Tachikoma, a popular character in the Ghost in the Shell universe [9]. By continuing to activate the browser extension, users can see Tachikoma in their browser. People who like the story or the character are motivated to install the browser extension and continue using it. We then prepared a web page where people can download and install the browser extension. We also advertised our data collection activities at large IT events. As can be seen in Figure 1, the current approach was successful from the standpoint of maintaining the number of active users, i.e., motivating users to continue using the browser extension.

6.3 Limitation of our dataset

The dataset we used has some limitations. When conducting further analysis, these limitations should be considered.

First, the demographics of the users of the browser extension are limited because we are distributing the browser extension to people who like the Tachikoma character. People who like Tachikoma tend to be familiar with IT; thus their use of the web does not necessarily accurately represent behavior of general web users. In addition, the browser extension and its distribution page are only available in Japanese; thus the data analysis will not reflect the behavior of global web users.

Second, depending on the purpose of analysis, the method used to label malicious URLs needs to be reviewed, . We used GSB entries to flag malicious URLs; however, this is not always desirable. Depending on the policies, users may implement different blacklists.

Third, the scale of the dataset is limited. Although the number of browser extension users was sufficient for the analysis in this paper, it is very small considering the number of web users in general. In particular, the number of victims is too small. When conducting other types of analysis, such as user classification, the scale of the data can be insufficient, and one such example is discussed in Section 6.4. To cope with this issue, measures such as effective campaigns to attract more users should be devised and deployed. Future works could consider these limitations of the dataset.

6.4 Further Analyses

This paper demonstrated the usefulness of analyzing user-side data collected through browsers. Further analyses are encouraged to deepen the understanding of user behaviors, including the analysis of access records before session reconstruction as discussed in Section 3.2. In this section, two more analysis directions are shown below.

We may use data that was unused in this paper to reconstruct hazardous paths in detail. For example, transition qualifiers, which can be collected through the chrome.webNavigation API, could be used to deepen the understanding of user behaviors. The API provides four transition qualifiers: "client_redirect," "server_redirect," "forward_back," and "from_address_bar." Their usability is demonstrated in a case, where a user browses pages in the following manner. (1) A user visits a search engine result page (whose transition type is set to "form_submit"). (2) The user clicks on one of the links on the page. (3) The user pushes the "back" button on the browser. In this case, the access record of the access (1) is used as the access record for the access (3), meaning that its transition type is "start_page" and the referer does not comply with access (2). In this study, the access path reconstruction scheme judges that the access (3) is the path entry point; however, we could trace back further by considering transition qualifier information.

Moreover, user behaviors can be analysed at a finer granularity. For example, user behaviors can be analyzed for each type of detected threats rather than using a binary label, i.e., malicious or not malicious. Indeed GSB provides types of detected threats, e.g., "MALWARE" and "SOCIAL_ENGINEERING"; however, we could not use these in this paper because our dataset did not have a sufficient amount of access records for each type of the threats detected by GSB. As discussed in Section 6.3, our dataset was too small to analyze accesses based on these types.

Various other analyses can be conducted for different purposes. These analyses will aid in building efficient schemes to improve user protection.

7 Related Work

Various studies have been reported in the area of malicious URL analysis. They take different approaches with different datasets. This section introduces major such works.

Previous studies have analyzed web page content to identify malicious sites [10–17]. A JavaScript code analysis at a bytecode level has been proposed [11] to cope with the obfuscation. Another study proposed a link structure analysis technique [12] to detect compromised websites by identifying structural anomalies. In addition, a cascading style sheets analysis technique has been proposed [13] to detect pages leading to malware downloads.

Lexical analysis has also been proposed to extract features from URL strings and identify malicious sites [18–23]. Among studies that apply lexical analysis, one study [19] attempts to achieve online learning; thus it does not use information that requires time to obtain. That study uses the URL string and host-related information, i.e., host name, primary domain, TLD, whois information, AS number, and geographical information, as features.

In addition, several studies have focused on building and analyzing redirection chains [24–29], which are often observed when users reach malicious URLs. For example, the SpiderWeb system [28] analyzes HTTP redirection chains. It uses five types of features, i.e., client, referer, landing page, final page, and redirection graph, to distinguish chains that correspond to malicious activity and those that are legitimate. WarningBird [29] detects malicious URLs posted on Twitter by analyzing the correlations of redirection chains, while Surf [25] identifies redirects to malicious URLs that are originated from search engine results.

Access paths followed by users who eventually fall victim to different types of malware download attacks, called malware download paths, have also been analyzed [30]. In that study, the authors proposed a download path traceback technique as well as a technique to determine whether the path is social engineering or drive-by, using various features, such as domain ages and the number of hops to exploit pages. Another study focuses on social engineering URLs [31]. That study uses a random forest algorithm to determine the occurrence of a social engineering attack from ad-related sites by learning about such past attacks using features extracted from the download paths. In addition, research focusing on identifying malicious exploit kits has also been reported [32].

Other studies have focused on user behavior [33–37]. A study analyzes traffic on a mobile cellular network to predict whether a user will visit a malicious URL within a month based on past browsing activities and a questionnaire [34]. The study also predicts whether a user would access a malicious URL within a session based on information from past records in the same session.

Various other studies have been reported in this area, including domain reputation systems [38–40] and signature generation techniques [41, 42]. Contrary to these, we collected data at the user side, which enabled us to analyze the user activities in detail by discerning users and browser tabs. We also proposed a preemptive domain filtering scheme that identified hazardous domains that were not included in blacklists.

8 Conclusion

Our user-side web access record collection approach enabled us to access to a wide range of data, such as user IDs, browser tab IDs, and user navigation information, which facilitated efficient and detailed analysis of user behavior. We have reconstructed hazardous paths from the collected data by continuously tracing previous access records until we identify entry points of the paths. The hazardous path reconstruction was efficient because it was able to discern users and browser tabs. Then, we analyzed the reconstructed hazardous paths to deepen the understanding of users' web browsing activities and revealed several analysis results, including that bookmarks are the major entry points of hazardous paths. It indicated that sanitizing bookmark entries will minimize the risk of accessing malicious URLs. Furthermore, we proposed a preemptive domain filtering scheme that identifies and filters domains that lead to malicious URLs. The effectiveness of the proposed scheme was demonstrated by revealing non-blacklisted domains that ultimately led users to malicious URLs. We hope that our work in this paper will contribute to the security of the web.

Acknowledgment

This study has been supported by WarpDrive project [2]. We thank all the colleagues of the project. We would like to extend our gratitude to our paper shepherd, Amin Kharraz, and the anonymous reviewers for their feedback and assistance.

References

[1] Google Safe Browsing. https://safebrowsing.google.com/. Accessed: June 1, 2020.

[2] WarpDrive. https://warpdrive-project.jp/. Accessed: June 1, 2020.

[3] Chrome: developer. https://developer.chrome.com/home. Accessed: June 1, 2020.

[4] chrome.webRequest. https://developer.chrome.com/extensions/webRequest. Accessed: June 1, 2020.

[5] chrome.webNavigation. https://developer.chrome.com/extensions/webNavigation. Accessed: June 1, 2020.

[6] chrome.history. https://developer.chrome.com/extensions/history. Accessed: June 1, 2020.

[7] Alexa top sites. https://www.alexa.com/topsites. Accessed: June 1, 2020.

[8] VirusTotal. https://www.virustotal.com/. Accessed: June 1, 2020.

[9] Ghost in the Shell. http://www.production-ig.com/contents/works_sp/16_/index.html. Accessed: June 1, 2020.

[10] D. Canali, M. Cova, G. Vigna, and C. Kruegel. Prophiler: A fast filter for the large-scale detection of malicious web pages. In *Proceedings of the 20th International Conference on World Wide Web*, 2011.

[11] Y. Fang, C. Huang, L. Liu, and M. Xue. Research on malicious javascript detection technology based on lstm. *IEEE Access*, 6, 2018.

[12] P. Ravi Kumar, P. Herbert Raj, and P. Jelciana. A framework to detect compromised websites using link structure anomalies. In *Computational Intelligence in Information Systems*, 2019.

[13] B. Chen, and Y. Shi. Malicious hidden redirect attack web page detection based on css features. In *2018 IEEE 4th International Conference on Computer and Communications*, 2018.

[14] B. Altay, T. Dokeroglu, and A. Cosar. Context-sensitive and keyword density-based supervised machine learning techniques for malicious webpage detection. *Soft Comput.*, 23, 2019.

[15] A. Fass, R. P. Krawczyk, M. Backes, and B. Stock. Jast: Fully syntactic detection of malicious (obfuscated) javascript. In *Detection of Intrusions and Malware, and Vulnerability Assessment*, 2018.

[16] M. Cova, C. Kruegel, and G. Vigna. Detection and analysis of drive-by-download attacks and malicious javascript code. In *Proceedings of the 19th International Conference on World Wide Web*, 2010.

[17] K. Rieck, T. Krueger, and A. Dewald. Cujo: Efficient detection and prevention of drive-by-download attacks. In *Proceedings of the 26th Annual Computer Security Applications Conference*, 2010.

[18] M. Darling, G. Heileman, G. Gressel, A. Ashok, and P. Poornachandran. A lexical approach for classifying malicious urls. In *2015 International Conference on High Performance Computing Simulation*, 2015.

[19] J. Ma, L. K. Saul, S. Savage, and G. M. Voelker. Learning to detect malicious urls. *ACM Trans. Intell. Syst. Technol.*, 2, 2011.

[20] R. Verma, and K. Dyer. On the character of phishing urls: Accurate and robust statistical learning classifiers. In *Proceedings of the 5th ACM Conference on Data and Application Security and Privacy*, 2015.

[21] A. Le, A. Markopoulou, and M. Faloutsos. Phishdef: Url names say it all. In *2011 Proceedings IEEE INFOCOM*, 2011.

[22] D. Huang, K. Xu, and J. Pei. Malicious url detection by dynamically mining patterns without pre-defined elements. *World Wide Web*, 17, 2014.

[23] G. Tan, P. Zhang, Q. Liu, X. Liu, C. Zhu, and L. Guo. Malfilter: A lightweight real-time malicious url filtering system in large-scale networks. In *2018 IEEE Intl Conf on Parallel Distributed Processing with Applications, Ubiquitous Computing Communications, Big Data Cloud Computing, Social Computing Networking, Sustainable Computing Communications*, 2018.

[24] Z. Li, S. Alrwais, Y. Xie, F. Yu, and X. Wang. Finding the linchpins of the dark web: a study on topologically dedicated hosts on malicious web infrastructures. In *2013 IEEE Symposium on Security and Privacy*, 2013.

[25] L. Lu, R. Perdisci, and W. Lee. Surf: Detecting and measuring search poisoning. In *Proceedings of the 18th ACM Conference on Computer and Communications Security*, 2011.

[26] H. Mekky, R. Torres, Z. Zhang, S. Saha, and A. Nucci. Detecting malicious http redirections using trees of user browsing activity. In *IEEE Conference on Computer Communications*, 2014.

[27] Z. Li, K. Zhang, Y. Xie, F. Yu, and X. Wang. Knowing your enemy: Understanding and detecting malicious web advertising. In *Proceedings of the 2012 ACM Conference on Computer and Communications Security*, 2012.

[28] G. Stringhini, C. Kruegel, and G. Vigna. Shady paths: Leveraging surfing crowds to detect malicious web pages. In *Proceedings of the 2013 ACM SIGSAC Conference on Computer & Communications Security*, 2013.

[29] S. Lee, and J. Kim. Warningbird: A near real-time detection system for suspicious urls in twitter stream. *IEEE Transactions on Dependable and Secure Computing*, 10, 2013.

[30] T. Nelms, R. Perdisci, M. Antonakakis, and M. Ahamad. Webwitness: Investigating, categorizing, and mitigating malware download paths. In *24th USENIX Security Symposium*, 2015.

[31] T. Nelms, R. Perdisci, M. Antonakakis, and M. Ahamad. Towards measuring and mitigating social engineering software download attacks. In *25th USENIX Security Symposium*, 2016.

[32] T. Taylor, X. Hu, T. Wang, J. Jang, M. P. Stoecklin, F. Monrose, and R. Sailer. Detecting malicious exploit kits using tree-based similarity searches. In *Proceedings of the Sixth ACM Conference on Data and Application Security and Privacy*, 2016.

[33] D. Canali, L. Bilge, and D. Balzarotti. On the effectiveness of risk prediction based on users browsing behavior. In *Proceedings of the 9th ACM Symposium on Information, Computer and Communications Security*, 2014.

[34] M. Sharif, J. Urakawa, N. Christin, A. Kubota, and A. Yamada. Predicting impending exposure to malicious content from user behavior. In *Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security*, 2018.

[35] F. L. Lévesque, J. M. Fernandez, and A. Somayaji. Risk prediction of malware victimization based on user behavior. In *2014 9th International Conference on Malicious and Unwanted Software: The Americas (MALWARE)*, 2014.

[36] M. Ovelgönne, T. Dumitraş, B. A. Prakash, V. S. Subrahmanian, and B. Wang. Understanding the relationship between human behavior and susceptibility to cyber attacks: A data-driven approach. *ACM Trans. Intell. Syst. Technol.*, 8, 2017.

[37] Y. Carlinet, L. Mé, H. Debar, and Y. Gourhant. Analysis of computer infection risk factors based on customer network usage. In *2008 Second International Conference on Emerging Security Information, Systems and Technologies*, 2008.

[38] M. Antonakakis, R. Perdisci, D. Dagon, W. Lee, and N. Feamster. Building a dynamic reputation system for dns. In *Proceedings of the 19th USENIX Conference on Security*, 2010.

[39] M. Antonakakis, R. Perdisci, W. Lee, N. Vasiloglou, and D. Dagon. Detecting malware domains at the upper dns hierarchy. In *Proceedings of the 20th USENIX Conference on Security*, 2011.

[40] M. A. Rajab, L. Ballard, N. Lutz, P. Mavrommatis, and N. Provos. CAMP: content-agnostic malware protection. In *20th Annual Network and Distributed System Security Symposium*, 2013.

[41] J. Zhang, C. Seifert, J. W. Stokes, and W. Lee. Arrow: Generating signatures to detect drive-by downloads. In *Proceedings of the 20th International Conference on World Wide Web*, 2011.

[42] J. Newsome, B. Karp, and D. Song. Polygraph: Automatically generating signatures for polymorphic worms. In *Proceedings of the 2005 IEEE Symposium on Security and Privacy*, 2005.