Accurate, Generalizable, and Practical Behavioral Models to Identify Impending User Exposure to Malicious Websites

JIN-DONG DONG, Carnegie Mellon University, USA
KYLE CRICHTON, Georgetown University, USA
AKIRA YAMADA, Kobe University, JP
YUKIKO SAWAYA, KDDI Research Inc., JP
LORRIE CRANOR, Carnegie Mellon University, USA
NICOLAS CHRISTIN, Carnegie Mellon University, USA

To keep users safe online, current protections frequently employ blocklists of known malware and phishing websites. However, such defenses suffer from an inherent gap between malicious content creation and its detection, leaving a window where users are left vulnerable. To address this limitation, earlier research has shown that one could use individual user web browsing behavior to identify imminent exposure to malicious content. While existing methods frequently rely on temporal proximity (e.g., aggregating browsing patterns over the recent past), they do not leverage temporal ordering in user browsing, which results in suboptimal performance and is, in practice, inadequate given the low base rates of malware incidence.

We introduce network and browser-level features (e.g., page rank, tab browsing time) and a temporal model that captures user behavior through a time-series representation. This not only improves classification performance by a significant margin (between 93% and 145% F1-score improvements) over previous models, but also maintains strong robustness across completely disparate sets of users. More importantly, our method shows strong resilience to concept drift, as performance holds steady over multiple years of testing. We discuss how this method is capable of anticipating future exposure. We also assess the relative importance of each feature to the performance, as well as their impact on false positive rates—whose minimization is critical to foster adoption. Finally, we discuss use cases for such behavior-based models.

CCS Concepts: • **Security and privacy → Software and application security**; • **Computing methodologies → Machine learning**.

Additional Key Words and Phrases: Web Security, User Behaviors, Sequence Modeling

ACM Reference Format:
Jin-Dong Dong, Kyle Crichton, Akira Yamada, Yukiko Sawaya, Lorrie Cranor, and Nicolas Christin. 2025. Accurate, Generalizable, and Practical Behavioral Models to Identify Impending User Exposure to Malicious Websites. *ACM Trans. Web* 1, 1, Article 1 (January 2025), 30 pages. https://doi.org/10.1145/3768587

1 Introduction

Traditional security protections for users browsing the internet often rely on blocklists. For example, Google Chrome checks the URLs the user is trying to access against the Google Safe Browsing (GSB) blocklist [20], which contains a list of known phishing and malware sites, and alerts the user that they are about to be exposed when there is a match. However, the blocklist approach suffers

Authors' Contact Information: Jin-Dong Dong, jd0@cmu.edu, Carnegie Mellon University, Pittsburgh, PA, USA; Kyle Crichton, kyle.crichton@georgetown.edu, Georgetown University, Washington, DC, USA; Akira Yamada, akirayamada@people.kobe-u.ac.jp, Kobe University, Kobe, Hyogo, JP; Yukiko Sawaya, yu-sawaya@kddi-research.jp, KDDI Research Inc., Fujimino, Saitama, JP; Lorrie Cranor, lorrie@cmu.edu, Carnegie Mellon University, Pittsburgh, PA, USA; Nicolas Christin, nicolasc@cmu.edu, Carnegie Mellon University, Pittsburgh, PA, USA.

ACM 1559-114X/2025/1-ART1
https://doi.org/10.1145/3768587

from several limitations, including: 1) There is an inherent time lag between malicious content creation and its detection, 2) Blocklists require constant maintenance, and 3) Attackers are able to bypass the detection with minimal cost (e.g., changing the URL of the malicious site). While other security measures exist, they are often either reactive (intervening after exposure), or use heuristics to detect malicious content, that either rely on the ever-changing characteristics of the website, content, or of the attack itself [1, 21, 22, 26, 42–44, 53, 54, 71, 74, 77, 79].

Recent studies have proposed an alternative approach to detect malicious content exposures: identify user behavior patterns [11, 58] instead of relying on characteristics of the threats themselves (e.g., URL). These systems are essentially binary classifiers where the input is a feature representation of user behavior (e.g., aggregated statistics on browsing activities over a period of time [58]) and the output is a variable denoting whether this behavior will lead to exposure. While user-behavior-based systems [11, 58] have demonstrated the feasibility of this approach, they have several critical limitations. First, these systems suffer from mediocre detection performance in terms of a high false-positive rate, i.e., they often classify that a user is attempting to visit a malicious site when in reality they are not. Second, behavior-based systems tend to assume fairly homogeneous populations (e.g., English-speaking users in Western countries)—their generalizability across different user populations is unknown. Third, we do not know how frequently behavior-based systems need to be updated (e.g., retrained) to achieve consistent performance over time. Fourth, the influence of user behavioral features, such as visit time or visited webpage category, on the detection performance is unknown, thus hindering the understanding of how these systems work.

We address these limitations in three ways: 1) We introduce a fundamental change to the representation of user browsing behavior. Previous studies mostly capture user behavior aggregated over a given interval (e.g., a browsing "session"), which fails to properly account for temporal ordering—going from page A to page B might be significantly riskier than going from B to A, if A is a search engine, and B a site distributing pirated software. We show that we can not only achieve significantly better performance with a representation that fully captures temporal information, but that we can also maintain strong robustness across long (multi-year) time intervals and vastly different user populations. In other words, we can successfully use data that is several years old to train classifiers that work properly today. 2) We introduce finer-grained features, at the browser level (e.g., tab browsing time), and the network level (e.g., page rank), and show how they markedly boost performance furthermore. 3) We evaluate feature importance to explain why the model makes certain decisions, and discuss the implications of an analysis of false positives and negatives. In addition, we evaluate the constraints, potential risks, and the best scenarios for the deployment of this proactive defense system.

We next describe the studies that inspired our work in Section 2. We then introduce the experiment design in Section 3 where we formulate the problem in Section 3.1, talk about the data that supplement our experiments in Section 3.2, how we engineer the features in Section 3.3, and our modeling choices in Section 3.4. Section 4 demonstrates the experiment results, which include performance enhancement (4.3, 4.4), system robustness across long (multi-year) time intervals (4.5), system robustness across different user populations (4.6), and the investigation of feature importance (4.7). and move on to discussing the limitations, potential risks, and the use cases in Section 5.

2 Related Work

We first discuss the security defenses that are commonly used to protect users browsing the internet (e.g., blocklists), including their effectiveness, and alternatives. We then shift toward work assessing the behavior of users in security, how to measure and model user web browsing, and how security outcomes are associated with particular behaviors.

2.1 Blocklists

Blocklists serve as the foundation of many security and privacy tools that help protect users. This includes lists of malware signatures in anti-virus software, wording of common scams in spam filters, and malicious website URLs in safe browsing and anti-phishing tools [80]. For example, Google Safe Browsing [20] is a blocklist embedded in Google Chrome that keeps records of malicious URLs and alerts users when they attempt to visit such a page. Privacy Enhancing Technologies, such as ad blockers, also rely heavily on blocklist-based solutions [38].

However, the blocklist approach suffers from two inherent problems. First, blocklists only protect users against known malicious threats. Existing estimates of the coverage of prominent URL blocklists vary widely, ranging from 21%–96% [37, 48, 78], and often contain little overlap [5, 41], indicating both a coverage and consistency problem. Second, even among identified threats, there exists a gap between creation and detection. Previous studies have estimated that the lag time ranges from several hours to several weeks [24, 48, 58, 60] and likely depends on the detection methods and the evasion techniques employed [48, 80]. Regardless of the exact delay, Nero et al. [45] demonstrates that the lag time is large enough for the perpetrator to profit off an attack before detection, and current methods of deterrence, like website takedowns, are not enough to compensate for that gap.

Making the detection problem more challenging, malicious actors have many evasion techniques at their disposal including cloaking [27], URL shortening [13], behavior-based evasion [48], and the use of compromised infrastructure [2]. These techniques are becoming more common, can delay or even prevent detection, and play a central role in most phishing attacks deployed at scale [49].

2.2 Threat-oriented security measures

Given these limitations, researchers have proposed numerous alternatives to supplement blocklists. A substantial body of work has been dedicated to identifying threats based on the content and the interactions with the resource, rather than purely on identifiers (URLs). These studies examine web page content [1, 71, 79], web traffic across a network [42, 53], or a combination of the two [31, 74] to classify whether a given web page is malicious or benign.

Several studies have used machine learning methods to develop content-agnostic malware detection [54], identify drive-by download attacks [56, 66], and detect malware distribution and download paths in large networks [26, 44, 77]. Taking this approach a step further, Soska and Christin used attributes of a website to predict whether it would become malicious sometime in the future [61], while Wu et al attempts to predict Network Security events [73].

While these methods provide much-needed help in bridging the detection gap, they also suffer from a scenario described as a game of "whack-a-mole" [41] or "cat-and-mouse" [48], that is, attackers can frequently deploy novel attacks while defensive measures constantly try to keep up.

2.3 Behavior-oriented security measures

Other than identifying threats based on the identifiers or features, another promising line of work focuses on the human aspect of security incidents. For example, one research direction analyzes and identifies malicious user behavior [3, 55]. Along the same lines, one study identifies the sociodemographic factors associated with risk-averse yet low cyber-security behavior [28]; others propose rigorous guidelines on how to conduct this type of studies [69]. Outside of the security realm, researchers have used browsing behavior to predict user demographic information accurately [25], mental health [46], and whether a user is a human or a bot [67]. In a security context, several studies have taken an epidemiological approach and identified key traits of users, their browsing patterns, and their machine configuration that are associated with security incidents [7, 8, 15, 33, 36, 40, 51].

Researchers studying user web browsing behavior have primarily taken one of two approaches to data collection: large-scale measurements of web requests captured over a network [30, 32, 35, 68] or smaller user-based studies drawing on sensors directly on the client machine [12, 39, 47, 65, 76]. While the former approach is easier to scale, the latter usually collects more accurate and highly detailed information [14]. Even though the datasets used in our study relied on the second approach, we were able to complement them with a large-scale collection (we will introduce our datasets in Section 3.2).

More closely related to our approach, Shen et al. used security events generated by a user's activity on their computer to predict what the next event would be with a precision of up to 0.93 [59]. In a similar study, Canali et al. leveraged logs of user browsing history to predict whether a given user would land on a malicious web page at any point over a 3-month period of observation with 87% accuracy [11]. Building on that work, Sharif et al. [58] developed a similar model to predict long-term exposure that produced comparable results. Even though the authors acknowledge that the false positive rate of the model might be too high for many practical use cases, we confirmed the feasibility of their short-term behavior-based security system in our study.

3 Experimental design

We define the problem and terminologies in Section 3.1, introduce the datasets, features, and models of choice in Sections 3.2, 3.3, and 3.4, respectively.

3.1 Problem formulation

The goal of our system is to detect whether specific patterns of online browsing behavior will lead to malicious content exposure. In a nutshell, our system is a binary classifier. The input to the model is a feature representation of a sequence of user browsing requests, and the model classifies whether this sequence will lead to exposure or not, a sequence classification task. This system is user-agnostic: we use a diverse set of sequences from different users to train the model. The sequence of user browsing requests considers both requests triggered by the user and by the website.

3.1.1 Threat model. This detection system can be deployed at the browser level (distributed to users) or network level (more centralized solution, e.g., deployed by Internet Service Providers, or ISPs). Browser-level deployment has the benefit of being more privacy-preserving, while network-level deployment can achieve faster detection and less computational overhead for users, but its effectiveness depends on the ability of the network provider (see discussion in 5.1). We confine the browsing behavior to be within a standalone browser (e.g., Google Chrome, Internet Explorer). Malicious content includes phishing and malware.

This system defends against attackers that are capable of hosting malicious content on the internet, and could bypass regular blocklist detections by, for instance, changing malicious URLs quickly. Attackers that could manipulate users' browsers or devices, and/or generate fake user behavioral patterns at scale, and/or intercept data to create adversarial examples [19] are out of scope.

3.1.2 Defining exposure. We define an exposure as a user's request to visit a malicious URL. To identify a malicious URL, we rely on the daily snapshots of the Google Safe Browsing (GSB) blocklist, a prominent blocklist for phishing and malware URLs that has consistently performed well in previous blocklist evaluations [37, 48, 60]. Intuitively, if a user visits a URL that appeared in the same-day GSB snapshot, we flag the URL as malicious. Inherently, this definition of exposure based on GSB (or any other blocklists) is limited, as it remains unclear when exactly a URL that is flagged as being malicious first became malicious, i.e., zero-days, and previous studies have shown that

there is an inherent unclear delay in detecting malicious content with GSB [24, 48, 58, 60]. To mitigate this problem, we adopt a t-day threshold to extend the coverage of our exposure definition. That is, if a user visits a URL on day t, and the URL appears in the GSB snapshots within the next n days (GSB snapshots at day t, $t + 1$, $t + 2$, ..., $t + n$), we consider the URL to be malicious. If the URL was in the GSB snapshots, but beyond the t-day threshold, we treat the visits as unexposed. If the URL was in the GSB snapshots, but was removed prior to the user's visits, we also consider these visits unexposed. A smaller t threshold is likely conservative, which may result in malicious content that we are unable to identify (false negatives). At the same time, setting a larger threshold increases the risk of getting more false positives.

3.1.3 Defining a browsing session. We develop a notion of browsing *session* to denote a sequence of user browsing requests. The separation of each session is based on user inactivity. We mirror previous work [6, 58] in using 20-minute of inactivity as the cutoff between different browsing sessions. While other methods for separating sessions may be viable (e.g., using a shorter or longer duration), we stick to the 20 minutes cutoff for comparison purposes. An *exposed session* is where a user has requested to visit a malicious URL in the session. The browsing activities that occurred after the request to the malicious URL are excluded from the exposed session. It is possible that the user is exposed again during the excluded activities, but we focus only on the first exposure in our experiment to limit the potential impact on the behaviors from other security tools.

3.2 Datasets

The data used in this study were collected from two sources: Security Behavior Observatory (SBO), an academic research study conducted in the United States, and Security Toolbar Trace Data (ST), a browser security toolbar widely used in Japan. Data from longitudinal panel studies of internet users, like that of the SBO and ST measurements, is exceedingly rare [75]. Given this scarcity, these datasets were selected due to their relatively large scale, in the number of users and duration of observation, and the unique access that we had to the data. In the following sections, we will cover how these datasets were collected, how we processed the data to derive the high-level features, and compare the differences between the datasets.

3.2.1 Security Behavior Observatory. The SBO was a multi-year study of the security behavior of home computer users conducted between May 2015 and July 2019. The project was run by a research team at Carnegie Mellon University [18]. Recruitment used a variety of methods, the most prominent of which was a university research recruitment service. As a result, most participants were recruited from the Pittsburgh metropolitan area. Upon confirming eligibility and providing written consent, participants were guided through the process of installing a set of "sensors" monitoring various activities—browser, file system, networking, etc—on their computer. Participants received $30 for enrolling in the study and an additional $10 per month for the duration of their participation. Once enrolled in the study, participants could discontinue their participation at any time. The SBO study received non-exempt, expedited IRB approval from Carnegie Mellon University.

Over the four-year period, the SBO recruited 623 participants, who, on average, remained in the study for just under two years ($\mu = 1.76, \sigma = 1.05$). The number of daily active users fluctuated between 100 and 200 participants for the majority of the study. Our work leverages data from 278 of those users that span from March 2017 to July 2019. This subset was determined by the availability of daily GSB snapshots to determine visits to malicious URLs and users that had data from all sensors. Data collection was limited to web browsing from Google Chrome on Windows computers (desktop or laptop) that were primarily used at home.

The SBO dataset contains very detailed user activity information. Besides the websites a user visited, the SBO also collected data about the use of private browsing and could differentiate

Table 1. Demographics differences between the SBO dataset and that of the general population of the United States.

Demographic	SBO	US
Age 18–34	73.0%	23.3% [9]
Female	60.7%	50.8% [9]
Computer-Related Field	27.1%	2.3% [50]
Bachelor's Degree or Higher	58.9%	32.3% [10]

between multiple browser tabs. Outside of web browsing, the SBO sensors also captured the other applications (besides the web browser) used, user interactions with their mouse, when the user was logged in, and when the machine was powered down. Forget et al. provides a complete discussion of the SBO architecture [18]. Sharing of the data received a category 4 exemption for secondary research from the IRB at Carnegie Mellon University.

As Table 1 shows, the SBO dataset skews younger than the general US population, with most participants falling between the ages of 18 and 34 (73%). The sample also contains more women (61%), those who have either education or professional experience in a computer-related field (27%), and those who have received a bachelor's degree or higher (59%).

3.2.2 Security Toolbar Trace Data. The security toolbar was a tool provided to customers by a large security company in Japan.[1] Web service partners distributed the toolbar on behalf of the security company as part of their own services. Customers who subscribed to these services could opt in to use the security toolbar. Users were provided information about the data collected when downloading the toolbar software, and how it is used exclusively for research purposes, before being prompted to provide consent to continue. The security toolbar, once installed, provided security services to the client and collected information about web requests. This data was encrypted locally and sent back to the security company's servers. The toolbar was used exclusively with Microsoft Windows Internet Explorer on desktop or laptop computers. Until recently, IE was still required within many Japanese businesses and government agencies [29], and as such had a very large user base in Japan at the time of data collection. The research team obtained this data through an agreement with the security company, and the sharing of this data received a category 4 exemption for secondary research by the Institutional Review Board (IRB) at our academic institutions.

The ST dataset is very large, with over a million users. Our analysis focuses on the data collected between October 2018 and July 2019, and July 2020 and February 2021, which corresponds to the most recent set of data that overlaps in time with data collected through the SBO, and data collected at the height of the COVID-19 pandemic, when online activity was reportedly peaking. On average, the dataset contains 60M web requests per day and 170K daily users. Each record contains the URL visited, the associated timestamp of when the request occurred, and a randomly-assigned, unique id corresponding to the user that made the request. No additional demographic information was collected.

3.2.3 Differences Across Datasets. One key contribution of our work is its generalizability across two very different user samples. We also draw comparisons with the dataset used by Sharif et al. [58], which was built on *mobile* users. While these users were also located in Japan, we expect significant differences in browsing behavior between mobile and PC users. The two datasets used

[1]The security company, and its browser toolbar, cannot be referred to by name due to a non-disclosure agreement with the company.

Fig. 1. Cumulative distribution of website traffic ranks across the SBO and ST datasets. We bin all sites beyond 10M in the same "unranked" category, hence the jump on the right of the graph.

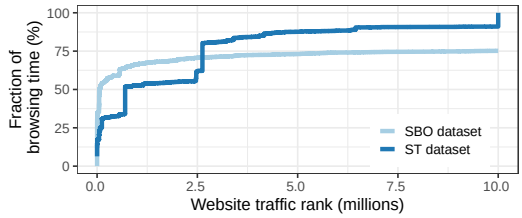
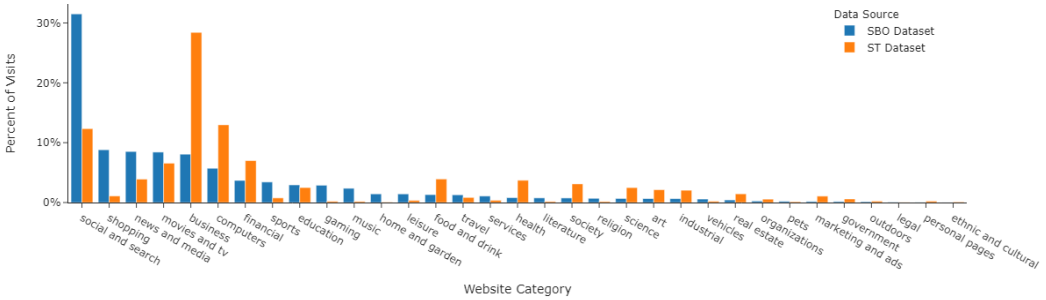


Fig. 2. Distribution of websites visited by category across the SBO and ST datasets. For comparison purposes, several anomalies were removed from the ST dataset that were related to automatically generated requests to advertising networks.



in this paper were collected by different types of entities (an academic institution and a private company), in two different countries with distinct cultures (the United States and Japan), with markedly different browsing patterns.

Specifically, we find that the rate of exposure is much higher in the SBO dataset and much lower in the ST dataset compared to previous findings. In the Sharif et al. study [58], 1 in every 999 browsing sessions contained a visit to a malicious URL (see Section 3.1.3 for the definition of a browsing session). In comparison, the exposure rate is 1 in every 166 sessions in the SBO dataset and 1 in every 3,324 sessions in the ST dataset. There are several competing hypotheses here.

One explanation is that global blocklists like the GSB are better at identifying malicious URLs visited by an English-speaking population compared to that visited by Japanese speakers. However, GSB is still the most widely used blocklist in Japan and we are not aware of the existence of a Japanese-specific blocklist of a similar scale and complexity. Furthermore, this would not explain the differences in exposure rates observed between the ST dataset and that used by Sharif et al., both of which were collected from Japanese participants. An alternative possibility is that participants in the SBO exhibit riskier browsing habits, possibly as a product of being younger than the older participants in ST.

In addition to exposure rates, we observe two major differences in the distribution of websites visited by participants in the two samples. First, browsing in the ST dataset skews towards websites with a lower traffic rank. Figure 1 shows that the cumulative distribution of visits across traffic rank is lower for the ST dataset than the SBO dataset for 60% of the traffic. This is likely a result of Japanese-language websites having a lower average traffic rank globally than their English-language

counterparts. Given that previous work has shown an association between low-traffic sites and malicious content [14, 58], we would expect this to dilute the predictive power of the traffic rank feature.

Second, as evidenced in Figure 2, we observe markedly different distributions in the website categories visited in the two samples. In the SBO dataset, participants tend to engage in more leisure activities such as visiting social media, searching, and shopping sites. In contrast, we observe a much greater proportion of visits to work-related websites such as business, computer, and financial sites in the ST dataset.

This could be due to ST protecting users from accessing malicious pages, which impacted the final exposure rate. However, we expect this impact to be minimal as we capture requests *before* ST kicks in, which should provide us with an unmodified user browsing behavioral sequence.[2]

3.2.4 Datasets limitations. There are two fundamental limitations to note before understanding how we processed and ran our experiments: 1) We do not preclude the possibility that SBO and ST participants used additional security tools in their browser or system along with our collection tools, or the fact that they could be using other browsers at the same time, which could impact coverage of. However, we expect the influence to be minimal: it is unlikely that a user would switch browsers in the *middle* of a browsing session; and security tools are generally reactive—i.e., they usually do not prevent exposure ahead of time, but mitigate the effects of an exposure. Ad-blockers and other privacy-enhancing technologies might block access to certain malicious resources proactively, but these resources need to already be in a blocklist, which, per our earlier discussion, still leaves an exploitation window open for the attacker – our longitudinal measurements allow us to uncover such effects. 2) Data are collected only on desktop and laptop computers. It is unclear how our results would translate to mobile devices. However, previous work using mobile data indicates that behavior-based prediction is feasible [58].

3.3 Feature engineering

We discuss how we engineer the high-level features for the downstream models. These include assigning a category for each web request, associating the pages with their traffic and PageRank score, advertisement/tracker identification, and assigning a category to each opened application (SBO only). These high-level features are incorporated with other low-level features, such as Session Depth, Session Time, to form our final feature vector for the downstream model (the complete list of the features can be found in Table 2, discussed in Section 3.4.3).

3.3.1 Categorizing Websites. As previous work has identified, certain categories of websites are associated with a higher risk of exposure to malicious content [33, 52, 58, 72]. To include this feature, we assign a category label to each website based on its domain. This process is done through a domain category classifier adopted from Crichton et al. [14], and the list of categories can be seen in Appendix B. For the SBO dataset, the 72-category taxonomy follows Crichton et al. [14]. For the ST dataset, which mostly contains Japanese-based websites, the same taxonomy did not perform well for our downstream tasks, and we condensed it to a more generalized version with only 49 categories.

3.3.2 Assigning Traffic Metrics. Expanding on the idea of extracting website information, we supplement each web request with the traffic rank and PageRank score of the website. We obtained this information from the Open PageRank project, which maintains a list of the ten million most popular domains. The traffic rank indicates the ordered ranking of the website, while the PageRank score approximates a search engine ranking score. Since Open PageRank's rankings are assigned

[2]The effect is minimal but not zero because ST's definition of malicious may not perfectly align with GSB's.

per website, not web page, we matched the dataset by domain name [17]. For some large websites, subdomains were included in the list separately (e.g., google.com and mail.google.com). If a matching subdomain was found, its traffic rank and PageRank were used.

3.3.3 Labeling Advertisements. In addition, we identify if the request was to an advertisement or tracker URL, as some malicious pages contain a significant number of advertisements [58]. We check each request against the same-day snapshot of EasyList rules, an open-source ad and tracker filter, to determine whether the URL would have been filtered on that day [4].

3.3.4 Categorizing Applications. The SBO dataset captures a host of additional system events besides browsing, including when a user switches between different applications on their computer. To make this information useful in our models, we assigned each application to one of 53 categories using the same methods and taxonomy as in previous work [14]. The categories are described in Appendix C.

3.4 Model and feature representations

We next discuss the models and the corresponding feature representation of browsing sessions. We first need to be able to (approximately) replicate the results of Sharif et al. [58] for a baseline comparison. Second, we introduce a new representation of the behavior features and the model of choices. Third, we introduce additional features, which provide a significant leap in improvement over the current state of the art.

3.4.1 Model for aggregated representation (baseline). The previous study conducted by Sharif et al. [58] employed an aggregated representation of a browsing session. This model of user behavior uses aggregated, session-level metrics to describe a user's behavior. For example, while each request in a browsing session is labeled by the website category the request was made to, the aggregate representation would simply be the total number of requests made to that specific category within the session This provided an aggregate distribution of website categories visited, but inherently results in the loss of information—particularly the order websites of different categories were visited in. The result of this is a 1-dimensional aggregated feature vector that represents a browsing session. To establish a performance baseline on our datasets, we use a three-layer Convolutional Neural Network, a close approximation to Sharif et al.'s model. The difference in model architecture between ours and that of Sharif et al. is the order of the layers connecting to the convolutional layers. We follow a more conventional approach to use a batch normalization layer and a Rectified Linear Unit (ReLU) between each convolutional layer [63, 64], and a linear classifier connected to the last convolutional layer.

We will refer to reported results from the Sharif et al. study as *Sharif NN-Baseline*. The baseline models we test on the ST and SBO datasets will be referred to as *ST NN-Baseline* and *SBO NN-Baseline*.

However, the features available in the SBO and ST datasets differ slightly from those of the previous study. The first difference is that neither the SBO nor the ST datasets captured the number of bytes transferred while the user was browsing. We also drop the six variables indicating the operating system (Android, iOS, or other) and browser (Chrome, Safari, or other) used in Sharif et al., since the ST (Windows, Internet Explorer) and the SBO (Windows, Google Chrome) datasets are each operating system- and browser-specific. Finally, our website categories (see Section 3.3.1 for details) differ from the 99 categories used by Sharif et al. since we do not have access to the paid service (Digital Arts) they employed to label their data.

3.4.2 Model for time-series representation. Instead of using a 1-dimensional aggregated feature vector to represent a browsing session, our approach retains the order of the sequence by leaving

the features of the requests disaggregated. This inherently preserves temporal information such as the time between each request, the order, and how users navigated between them. The result is a 2-dimensional feature matrix: the columns are the different features of each request, the rows are the requests.

There is an abundant choice of models for sequence classification, especially with the rise of transformer-based models [16]. However, to evaluate the effectiveness of the temporal information and minimize the performance impact from the model itself, we use a Long Short-Term Memory (LSTM) model for its simplicity and as a proof-of-concept. Our LSTM models feature a single layer of 100 hidden units, followed by a linear layer as the classifier. LSTM models using the same set of features as Sharif et al. [58], although not aggregated to the session level, are referred to as *ST LSTM-Basic* and *SBO LSTM-Basic*, where the *ST LSTM-Basic* model is trained on the ST dataset, and the *SBO LSTM-Basic* is trained on the SBO dataset.

3.4.3 Model for time-series representation with enhanced features. In addition to changing the model and the underlying representation of the browsing data, we also introduce a set of new features. We separate these features into two groups: network-level features and browser-level features. These groups represent the most likely deployment points for such a system, either on a network (i.e., an ISP or a company's private network) or within the web browser deployed directly on the user's machine. The closer the system is deployed to the user, the more data it inherently has access to. Browser-level features are only available in the SBO dataset, and therefore, the results are limited to that sample of data. As such, we test three models that we will refer to as ST LSTM-Network (trained on ST), SBO LSTM-Network, and SBO LSTM-Browser (trained on SBO).

The new features are summarized side by side with those from the LSTM-Basic model in Table 2. Features added to the LSTM-Network models include whether the URL was flagged as an advertisement by Easylist [4], its PageRank according to OpenPageRank [17], and whether the user has already visited the domain (not URL) earlier in the session. The browser model comprises a whole host of additional features available in the web browser, including which browser tab is being used, the number of other tabs and windows the user has open, and whether they are using private browsing mode. In addition, we include information about how long (time) and how many requests (depth) have been made since the user began actively using their web browser, and even the current browser tab. The model also incorporates browser-generated flags for web requests that are client or server redirects, and those generated by either the forward/back buttons or the address bar. The event type feature is a special case for the browser model as it contains web navigation events (e.g. link, typed, keyword generated, reload, etc.) and non-browser events (e.g. power on/off, login/logout, lock/unlock, and foreground application changes). Therefore, in addition to a richer feature set, the browser model also incorporates a greater number of events into the data sequence used by LSTM, as the basic and network models only have access to web request information. Finally, since the browser model captures changes in the application that the user is working with, we also include the application category for application change events. For a more detailed description of these features, refer to Appendix A.

In addition to the new features, we slightly modified the features used in the LSTM-basic model. First, we separated the website traffic rank into 10 bins of size 1 million each rather than relying on a single flag indicating whether the domain is in the top 10 million or not. This allows for greater differentiation between high-, medium-, and low-trafficked domains. Second, we modified the hour of the day to use a cyclic function rather than 24 binary flags. This allows the model to capture trends in browsing across the day, while also acknowledging that browsing at hour h and at hour $h + 1$ are unlikely to be very different from each other.

Table 2. Summary of features used in different LSTM models. Website Cateogory is composed of 72 categories for the SBO and 49 categories for ST

Feature	Basic	Network	Browser
Session Depth	✓	✓	✓
Session Time	✓	✓	✓
Weekend	✓	✓	✓
Hour (24 bin)	✓		
Hour (cyclic function)		✓	✓
Traffic Rank (bool)	✓	✓	✓
Traffic Rank (10 bin)		✓	✓
Website Category*	✓	✓	✓
Ad/Tracker filter		✓	✓
PageRank		✓	✓
New Domain in Session (bool)		✓	✓
Event Type			✓
Private Browsing			✓
Current Browser Tab			✓
Browser Tabs Open			✓
Browser Windows Open			✓
Active Browsing Depth			✓
Active Browsing Time			✓
Tab Browsing Depth			✓
Tab Browsing Time			✓
Server Redirect			✓
Client Redirect			✓
Forward/Back			✓
From Address Bar			✓
Application Category			✓

4 Experiment and results

In the following sections, we evaluate the performance of our models. Before diving into the results, we detail the hyperparameters used for our data and model, and discuss the evaluation metrics in Section 4.1. We then discuss experiments and results on each of these topics:

- Baseline comparison in Section 4.2
- Performance boosts by our features and models in Section 4.3 and Section 4.4
- Model robustness in Section 4.5 (robustness over time) and Section 4.6 (robustness across different datasets)
- Feature importance (through an ablation study) in Section 4.7
- Ability to anticipate future exposures in Section 4.8

Finally, we analyze the false positives and negatives in Section 4.9.

4.1 Hyperparameters and evaluation metrics

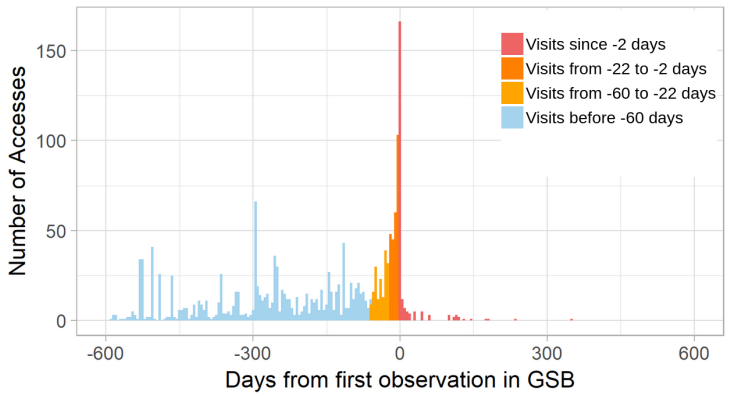
Table 3 summarizes the hyperparameters discussed in this section. We chose model architectures and training parameters to reproduce the previous work or based on conventional standards. We did not aggressively tune the parameters as we find our models robust and performing well without extensive parameter tuning.

4.1.1 t-day threshold. Figure 3 shows the distribution of visits in the SBO dataset relative to when a URL is first labeled as malicious by GSB. Working from the left of the distribution, we find a noticeable rise in visits starting about 60 days prior to being flagged by GSB, an acceleration in traffic growth around 22 days prior, and a spike within two days of being flagged. After a URL is labeled as malicious, at day zero, visits immediately drop to only scattered visits thereafter. This is similar to the trends observed by Sharif et al. at 38 (rather than 60), 22, and 2 days [58].

Table 3. Hyperparameters used in our experiments

	SBO		ST	
	NN	LSTM	NN	LSTM
Data				
t-day threshold	2		2	
Session cutoff	20 minutes		20 minutes	
Undersampling ratio	25:1		10:1	
Sequence length	-	100	-	100
Model architecture				
Layer type	1-d Conv	LSTM	1-d Conv	LSTM
Classifier type	Linear	Linear	Linear	Linear
Kernel size	5	–	5	–
Number of channels	128	–	128	–
Number of layers	3	1	3	1
Number of hidden units	–	100	–	100
Training parameters				
Loss function	Binary cross-entropy		Binary cross-entropy	
Optimizer	Adam		Adam	
Learning rate	10^{-3}		10^{-3}	

Fig. 3. SBO visits to URLs relative to when they first appeared in GSB. Negative values indicate the user visited the URL prior to it being flagged, positive values indicate visits after being flagged.



Based on the approach taken in the previous study and our experiments on various (2-, 22-, and 60-day) thresholds, we settle on the 2-day threshold ($t = 2$) for our main experiments, where we scored an ROC AUC of 0.990 and an F1 score of 0.830 for ST LSTM-Network. In comparison, the same model achieves an ROC AUC of 0.991 and an F1 score of 0.795 with a 22-day threshold, and an ROC AUC of 0.890 and an F1 score of 0.778 with a 60-day threshold. Similarly, Our best model, SBO LSTM-Browser, reaches an ROC AUC of 0.998 and an F1 score of 0.908 with a 2-day threshold, an ROC AUC of 0.993 and an F1 score of 0.824 with a 22-day threshold, and an ROC AUC of 0.993 and an F1 score of 0.831 with a 60-day threshold. Indeed, the larger the threshold, the more diverse the behaviors are in the additional exposed session, which makes it slightly harder for our model to classify them correctly.

4.1.2 Data split and sequence preparation. To deal with the imbalance between the numbers of exposed and unexposed sessions, we undersample the unexposed sessions at a 25:1 ratio with

the exposed sessions in the SBO dataset, and at a 10:1 ratio in the ST dataset where samples are abundant. We choose undersampling to make the training of the classification computationally feasible since the original exposure rate is extremely low (see Section 3.2.3). We do not find that undersampling has a significant impact on the loss of information for the majority class, as we obtained similar results when undersampling with different random seeds. We also adjust the weights for each class during training according to the number of examples in the class. The training and test sets are assigned at random, maintaining an 80-20 split.

We use the last 100 requests leading up to, and including, either 1) the last URL visited in an unexposed session or 2) the first malicious URL visited in an exposed session. We ignore sessions that have fewer than 5 requests. We tested sequence lengths of 50, 100, 150, and 200, and 100 requests proved to yield optimal performance. We hypothesize that 100 requests provide enough information for the model to understand browsing history without introducing too much noise from early, less-relevant activity.

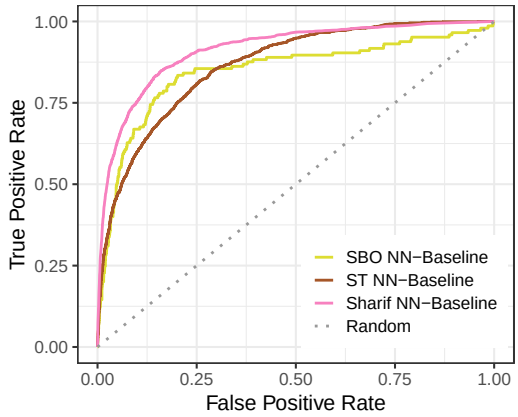
4.1.3 Evaluation metrics. We evaluate our binary classifiers primarily with two metrics: 1) the Receiver Operating Characteristic (ROC) curve and its Area Under Curve (AUC) score, and 2) the Precision-Recall curve and the F1 score, the harmonic mean of precision and recall. Since a binary classifier outputs an arbitrary real value, a threshold value, i.e., the confidence threshold, or the decision boundary, between two classes determines the outcome. The ROC curve plots the True Positive Rate (TPR, or sensitivity) against the False Positive Rate (FPR, or 1-specificity) at different threshold values, which provides excellent information for evaluating the trade-offs in different scenarios (whether we want to maximize TPR or minimize FPR). However, ROC results only show half of the picture, as they do not convey information about the precision or the negative predictive value, and the performance on imbalanced datasets may appear to be better than it actually is by looking at only TPR and FPR.

As a very rough average over the three datasets we looked at (SBO, ST, Sharif et al.), 1 in 1,000 browsing sessions overall results in an exposure. If we design a system with a true positive rate of 95% and a false positive rate of 5%, and this system processes 1,000,000 sessions in a day, we would expect 1,000 sessions to be truly malicious. Our system would catch 950 of them, which is good, but would also produce 0.05×999,000=49,950 false alarms, which would completely drown out the true positives. To have only *half* of the total number of alarms be valid, we would need to bring that false positive rate to roughly 0.1% (for the same true positive rate of 95%). Therefore, we complement ROC curves with Precision-Recall metrics, which focus on a different type of error [57] by ignoring True Negatives completely.

4.2 Result: Aggregated representation (baseline)

We first compare the results from our NN model trained on the baseline feature set in an aggregated representation for each dataset, SBO NN-Baseline and ST NN-Baseline, to the results in Sharif et al. [58]. Figure 4 displays the Receiver Operating Curves (ROC) for SBO NN-Baseline, ST NN-Baseline, and the results reported by Sharif et al. The y-axis represents the percentage of exposed browsing sessions correctly identified, the true positive rate, and the x-axis displays the percentage of unexposed browsing sessions incorrectly flagged as exposed, the false positive rate. The ROC and the area under the ROC (AUC) metrics are commonly used to evaluate classifier performance. The curves in the figure show that we can reproduce similar (albeit slightly worse) results to those achieved by Sharif et al. (AUC: 0.913). The SBO NN-Baseline achieves an AUC score of 0.847, and the ST NN-Baseline produces an AUC score of 0.853. The difference is likely caused by our not having access to some of the features Sharif et al. relied on, notably the number of bytes transferred with each request.

Fig. 4. ROC Curves comparing the performance of NN run on SBO, ST, and that reported by Sharif et al.



The F1 scores for SBO NN-Baseline and ST NN-Baseline are subpar, they are 0.370 and 0.422, respectively. Sharif et al. did not report F1 scores, but having had a chance to look at their data, they were equally low, if not worse. This initial result motivated us to develop better feature representations and extract additional features for the classification system to be practical.

4.3 Result: Time-series representation

Figure 5 compares the ROC curves for the ST and SBO LSTM-Basic model, which employ a time-series representation of browsing, to the SBO, ST, and Sharif NN-Baseline. As the figure illustrates, the LSTM model performs substantially better than the NN-Baseline on both datasets. Specifically, SBO LSTM-Basic, in the darker blue, achieves an AUC of 0.967 and an F1 score of 0.714, respectively, a 14.2% and 93.0% increase over the SBO NN-Baseline. Similarly, the ST LSTM-Basic, in darker green, reaches an AUC of 0.963, and an F1 score of 0.673, a 12.9% and 59.5% increase over the ST NN-baseline. Since the LSTM-Basic model employs the same set of features as the NN-Baseline, just in a disaggregated format, the difference in performance can be solely attributed to how the underlying browsing behavior is represented. This result shows that preserving the temporal information, e.g., the order in which browsing requests occur, is crucial to this task.

4.4 Result: Time-series representation with enhanced features

The addition of these new features raises model performance to even more desirable levels (AUC above 0.98). Figure 6a shows the ROC curves for the LSTM-Baseline, LSTM-Network, and LSTM-Browser models trained on SBO and ST. SBO LSTM-Network achieves an AUC of 0.995 and an F1 score of 0.847, ST LSTM-Network achieves an AUC of 0.990 and an F1 score of 0.830, both significantly outperform the corresponding LSTM-Baseline and NN-Baseline models. The LSTM-Browser model, only available using the SBO dataset, achieves an even higher performance with an AUC score of 0.998 and an F1 score of 0.908. This represents a 17.8% increase in the AUC score and a massive 145.4% increase in the F1 score over SBO NN-Baseline.

These improvements are quite stark when examining performance with low false positive rates. Figure 6b shows the same curves as before, but limited to false positive rates between 0% and 5%. In particular, the SBO LSTM-Browser model can achieve true positive rates of 88% and 92% with false positive rates of 0.05% and 1% respectively. Comparing to the previous state-of-the-art [58], the numbers are 56% TPR at 3% FPR without calibrations from external tools.

Fig. 5. ROC Curves of NN and LSTM models with the baseline feature set.

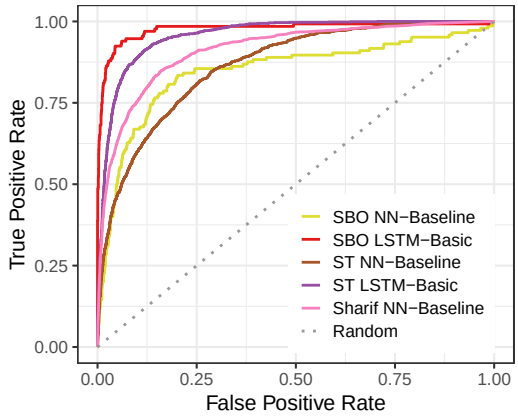
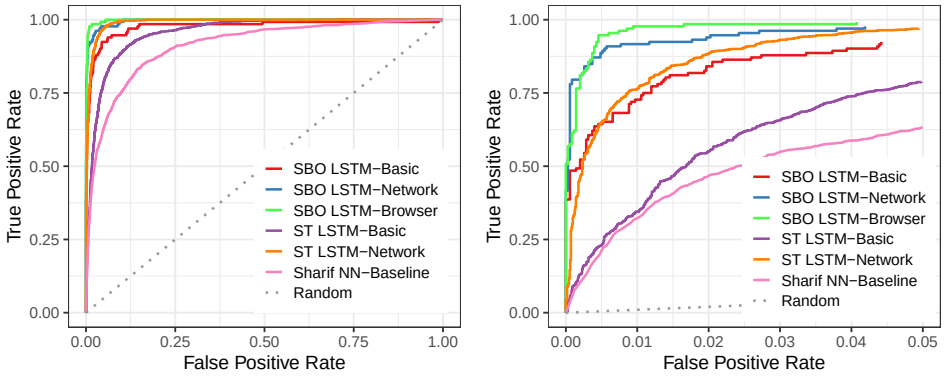


Fig. 6. ROC Curves for LSTM with additional feature sets (Network/Browser features).



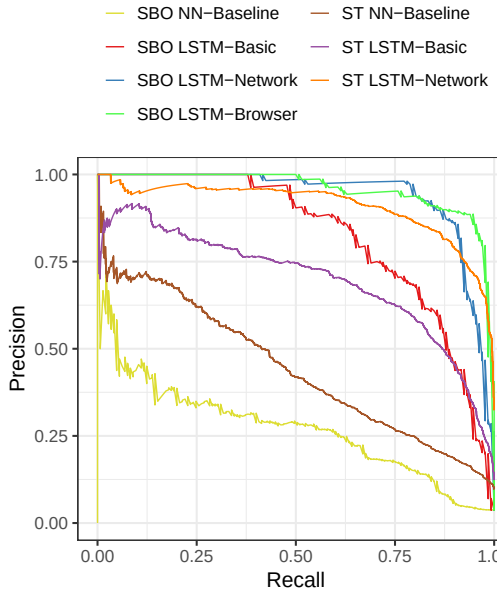
(a) False positive rates 0%–100%

(b) False positive rates 0%–5%.

While Figure 6a does not show major differences in the AUC score between SBO models using network-level (AUC: 0.995) and browser-level (AUC: 0.998) features, the features have a substantial impact on precision and recall. As Figure 6b illustrates, this difference is most pronounced at low false-positive rates, where the LSTM-Browser model can recall (i.e., correctly identify) 56% of exposures with near-perfect precision (i.e., almost no false positives) as compared to only 39% in the LSTM-Network model.

The precision and recall curves for all of our models are shown in Figure 7. The performance of the models with features in the time-series representation, regardless of the feature set, outperforms the models with features in the aggregated representation by a large margin. The SBO LSTM-Browser model's F1 score of 0.908 represents a marked improvement over the SBO LSTM-Network model's F1 score of 0.847.

Fig. 7. Precision and recall curves for all the models.



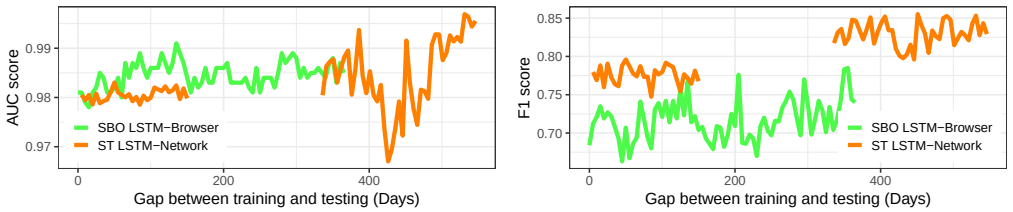
4.5 Result: Model robustness across long time intervals

We already showed that the model with the time-series representation exhibits spatial robustness, i.e., it performs well on fairly different datasets in SBO and ST. We want to further evaluate if the model has temporal robustness, i.e., whether we can still successfully identify exposures even when the model is trained on much older data compared to the time of the exposure. For gaps within a year, we design a train/test data split to use browsing sessions that occurred towards the end of our sample (SBO: last 3 months, ST: last month) as test set, and sessions occurring during a moving window of time before that (SBO: 9 months, ST: 4 months) as the training set. For gaps with more than a year in the case of the ST dataset, we train the model with data from October 2018 to July 2019 and test it on data in 2020 and 2021. As shown in Figure 8(a), the AUC scores are consistently between 0.978 and 0.991 as the gap in time increases, and we do not observe a downward trend. Figure 8(b) confirms this: if anything, the F1 score appears to slightly increase when the time gap increases (which we attribute to using more data for training: 10 months compared to 4 months). This shows that behavioral indicators of exposure are robust with respect to time, and frequent retraining to maintain performance is not required. This is extremely important: even though our data is a few years old, user behavior changes sufficiently slowly that past data remain highly relevant for training current models.

4.6 Result: Model robustness across different user populations

In addition to showing that the models are robust against concept drift in time, we also demonstrate that the models can perform well when running on diverse samples of users. As discussed, the participants in the SBO and ST datasets come from different countries with their own distinct culture (the United States and Japan), have different rates of exposure, and exhibit distinct browsing patterns (see Section 3.2.3) In this experiment, we use the ST and SBO LSTM-Network models that were on one dataset, but instead test them on the other. To do this, we had to create a one-to-one

Fig. 8. AUC and F1 score of the model as the time gap between training and test sets increases. (Since we are considering two separate periods for ST, we do not have data for gaps 150–366 days.)



(a) AUC (b) F1 score

mapping between the 49 ST website categories and the 72 SBO categories (see Appendix B for details).

We find that the ST-train/SBO-test model achieves a very high ROC AUC of 0.994 and F1 score of 0.843. Meanwhile, the SBO-train/ST-test model achieves an ROC AUC of 0.886 and an F1 score of 0.435. The weaker performance of the SBO-trained model, particularly in precision and recall, is likely a result of two factors. First, in mapping the SBO website categories (72) to the ST categories (49), we reduce the number of features, inherently losing information. However, in the reverse direction, this is not the case. We will discuss the importance of the website category feature further in Section 4.7. Second, in training on the SBO and testing on the ST dataset, we are extrapolating from a much smaller number of observations to a much larger set. As a result, we are not surprised to see lower performance from the SBO-train/ST-test experiment. However, it should be noted that the performance is still slightly better than that achieved by the NN-Baseline models. Furthermore, the high level of performance achieved in the ST-train/SBO-test experiment demonstrates that user browsing behaviors that are associated with exposure are highly similar across datasets despite the aforementioned differences between the two samples. This further demonstrates the robustness of our behavior-based models and indicates that behavioral indicators of exposure are generalizable.

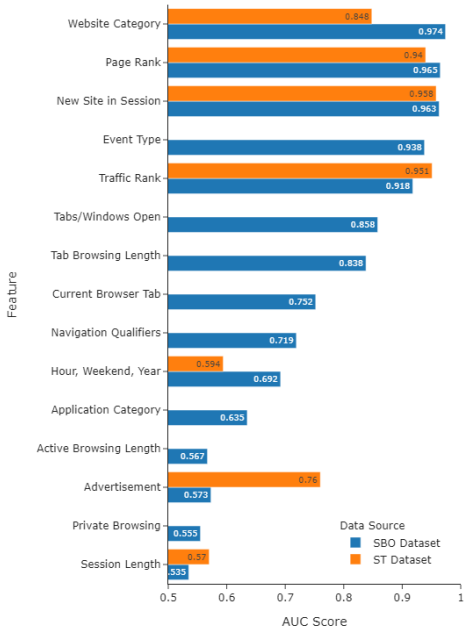
4.7 Result: Understanding feature importance

Understanding how the models correctly identify exposures is equally important to developing systems with high precision. We run an ablation study by measuring how performance changes when specific features are removed. In particular, we train the best-performing models for both datasets (i.e., SBO LSTM-Browser, ST LSTM-Network) with all but one feature removed. The result can be seen in Figure 9. When a feature is not available in the ST dataset, which is the case for all browser-level features, the bar is left blank.

Features that are important in both SBO and ST include page rank, traffic rank, and whether the request is to a new (unseen) website in the session. The page and traffic rank features are among the most significant features in both datasets. We find that exposed sessions in both datasets typically include more low-ranked or unranked websites, especially near the point of exposure. As a result, page and traffic rank features are strong indicators of exposure. In addition, whether the request is to a new website in the session is also an important feature. The reason is that most points of exposure arise when users traverse to a new domain, rather than ending up on a malicious page from other parts of the same website.

There are also features that display different significance between the datasets. The largest difference comes from the advertisement feature. Overall, we observe very few URLs labeled as ads in the SBO dataset and a large proportion of them in the ST dataset. The sparse distribution of ads in the SBO dataset likely prevented the model from identifying as many related trends,

Fig. 9. Model performance with individual features. Blank fields indicate that the feature was not available.

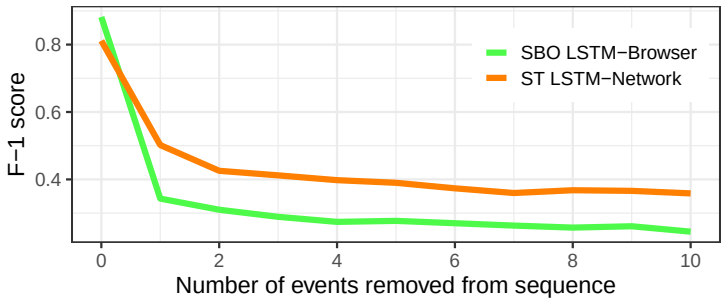


thus diminishing the significance of this feature. While the lack of ads in the SBO may be a result of users visiting different types of websites, we believe this is more likely a result of the SBO capturing data through the browser, which may filter or suppress some of these requests that would otherwise be captured over the network. On the other hand, the website category contributes more to the SBO dataset. One explanation is that the SBO dataset simply employs a greater number of categories, thereby providing slightly more granular information. Alternatively, it could be that the website categories in the ST dataset are not as accurate since the category classification model was trained on a predominantly English-language corpus of websites but asked to classify a mostly Japanese-language one.

4.8 Result: Anticipating future exposure

Although we are able to identify imminent exposure, we find that it is inherently difficult to anticipate future exposures in a session. To confirm this, we ran a series of experiments that removed successive requests from the end of the sequences. In exposed sessions, this meant removing the web request to the malicious URL and those immediately preceding it. Figure 10 shows the model's performance over the number of requests removed from the end of the sequence. Even though using the full sequence (100 requests) yields the best performance, we can see that the last 5 requests in the sequence contribute the most to exposure anticipation compared to other, earlier requests, and removing the last request in the sequence leads to the largest decrease in performance. The final request typically has very distinct features in exposed sessions, which are not previously present in the user's browsing (e.g., the page has not been visited before, it is unranked, and its category is new). In short, one *can* anticipate future exposure throughout a

Fig. 10. F1 score of the model as we remove successive requests from the end of the input sequence.



browsing session, but without the same precision. As such, anticipation, as opposed to identification, is more speculative in nature. Due to the base rate fallacy discussed earlier, this makes the use of anticipations in a system that actively intervenes to prevent exposure (e.g., by blocking) several pages ahead of time relatively difficult. However, such anticipations may be relevant for use cases where one wants to provide feedback to a user about the relative risk as they browse the web, and, as we demonstrated, a model that intervenes right when the user is about to connect to the exposed page is highly viable. This experiment also illustrates one of the primary advantages of the sequential model: it can differentiate between the most recent requests and those further back in the session's browsing history.

4.9 False positives and negatives

Just as essential as discerning how the model correctly identifies exposures is understanding the factors driving incorrect classifications. A classic machine learning problem for classification is when the model outputs an inference score close to the decision boundary, which is often regarded as the model being *not* confident in its classification. From the distribution of these scores, we find that the model confidently assigns 96% of unexposed sessions a score less than 0.01 and 77% of exposed sessions a score greater than 0.95. The sessions that fall between are the cases where the model cannot make as clear a distinction and may result in false positives and negatives. We define the two original sets ($\hat{y} < 0.01$ and $\hat{y} > 0.95$) as thresholds for what the model confidently thinks is benign or malicious, and we compare the distribution of features from those sets to that of exposed and unexposed sessions that fall between to identify the drivers of false positives and negatives.

We separate the features into three groups: (1) features that are drivers of both false positives and negatives, (2) features positively associated with exposure that lead to false positives, and (3) features negatively associated with exposure that influence false negatives. In the first group, those driving both false positives and negatives, includes domains labeled as streaming, advertising, and ad redirects. The model strongly associates web requests to these domains with exposure. Exposed sessions with a low proportion of these requests were often given a low $\hat{y}$, and unexposed sessions with a high proportion were frequently given a high $\hat{y}$. In both cases, this would typically lead to incorrect classifications. Similarly, the model associates a higher proportion of requests to pages ranked in the top 1M websites with unexposed sessions and requests to pages ranked higher than 3M with exposure. Sessions that went against this trend were typically mislabeled.

The second group of features, those driving false positives, consists of visits to URLs offering rewards and freebies, server-redirected requests, the creation of new browser tabs, and switching between tabs. The model strongly associates a higher frequency of these requests with exposure.

As such, unexposed sessions that contained a higher proportion of these requests were more likely to receive a higher $\hat{y}$, contributing to the generation of false positives. However, we did not observe the reverse trend affecting false negatives.

The third set of features, those driving false negatives, includes nine website categories that are associated with unexposed sessions. These categories include news, education, healthcare, finance, legal, retail, industrial, organizations, and religion. Exposed sessions containing a higher proportion of requests to these types of URLs were likely classified as being benign. Like before, we did not observe the reverse trend affecting false positives.

5 Discussion

Our results demonstrate that, with the right representation, behavior-based classification models have the potential to become a viable tool for protecting users online. In this section, we discuss the limitations and possible risks of our research, how to position this technology for real-world use cases, and propose future work that would move this technology forward.

5.1 Limitations

There are several limitations to our findings that are vital to understand before applying these techniques in real-world scenarios. First, user web-browsing behavior can change due to the introduction of generational technologies. For example, the introduction of multiple browsing tabs in the late 1990s changed how people browsed the web over the course of the next two decades [14]. As such, these models will still need to adjust over time, albeit slowly. However, accounting for changes in browsing behavior over the years is much more preferable than having to adapt to changes in the threat landscape (e.g., malware signatures), which can evolve hour by hour. Second, just like any other machine learning solution, out-of-distribution instances are hard to correctly classify, as discussed in Section 4.9. This is sometimes overcome by additional human supervision to make the final decision. Third, the effectiveness of network-level deployment depends on the ability of the network provider. For regular home users, the ISP doesn't know the full URLs with HTTPS traffic, so the signal to the model could be weaker. However, this would not be a problem for network providers with HTTPS inspection capability, such as in private organizations/companies.

5.2 Possible Risks

Our findings also raise some concerns regarding possible deployment risks and adversarial use cases.

First, as with any application that collects personal data, there is a risk of leaking that information to unwanted parties. Information contained in URLs, or even in domain names, can reveal sensitive information about those who visit them. This creates the potential for privacy harm if there is a breach.

Second, these models open up additional risks of side-channel attacks that could allow third parties to infer information about the user. Previous work has demonstrated that eavesdroppers on a network can identify the web pages a user visits by observing timing, delays, and packet sizes in HTTPS traffic [34, 62]. If processing locally, these models will add additional delay to outgoing web requests, which could provide further information about what pages the user is visiting and their level of risk. If run over a network, the size and timing of packets returned to the user's machine from the system host may also allow observers to infer this kind of information. These side channels could further aid third parties in tracking user browsing, fingerprinting the user, and assigning risk profiles to individuals.

Third, there are several adversarial use cases where a behavior-based model could be abused. For example, suppose our model is used to identify high-risk users within an organization to allocate

more security resources or training. The same tactics could be employed to penalize these same users. On a broader scale, it is not much of a stretch to envision an individual's risk profile being compiled and then used to determine access to certain services, charge them different prices, or assess fault in the event of a breach. In another case, behavior-based classification could be used to aid perpetrators of censorship. By redefining "malicious content" to include sites that a censor deems unsavory, our methods could be used to better prevent access to these sites and identify new content to block. However, this assumes there are common patterns to visiting these types of sites that behavior-based models could pick up on. This likely depends on the type of content the censor is aiming to block, but further research is warranted. In a final adversarial case, we believe that it is likely that behavior-based models could be used to fingerprint and identify users across browsing sessions. Previous work has shown that a user's browsing is fairly unique and does not conform to the concept of an average user or groups of users [14, 70]. Using similar methods to our own, it is possible that third parties that can observe web traffic on a large scale, like advertisement networks or internet service providers, could identify users without the use of other tracking tools. This raises significant privacy concerns that should be investigated further.

5.3 Use Cases in Security Applications

With these limitations and risks in mind, we now turn towards possible security applications where these methods could be deployed. The differences in performance of our models indicate that different approaches will be more useful for certain applications than others. We break these use cases down into four primary areas: 1) those that directly intervene in the course of a user's browsing, 2) integration into other security tools, 3) deployment across an organization's network, and 4) individualized models for users of different risk tolerance levels.

The first set of use cases is those that interact directly with the user to protect them as they browse. With each new web page a user visited, such a tool would monitor a user's behavior and classify requests as they were made. This information could be presented to the user as they browse, providing continuous feedback on their level of risk. Alternatively, the tool could operate in the background and, upon reaching a specified threshold, trigger an intervention. Since a user-facing system like this would require a low false positive rate, our results indicate that it would be better deployed locally in the user's browser.

However, as we briefly discussed in Section 4.2, this system is only practical when it can achieve an extremely low false positive rate. The level of performance our best model, SBO LSTM-Browser, achieved (92% TPR at 1% FPR) borders on the stringent conditions that would make this system feasible for general use.

This leads to the second set of applications that, even if the accuracy of the model does not meet the more stringent thresholds, could still provide an immediate security benefit. In these cases, feedback from behavior-based models could be integrated into other security measures like anti-virus software. Armed with a real-time indicator of a user's browsing risk, these tools could modulate their own behavior, ramping up security measures in risky situations and lowering them as users returned to safer territory. This would enable tools to provide the same services while consuming fewer computational resources and interfering less with the user's operation of their device. Another application of behavior-based classification is to supplement malicious website detection, as positive classification by the model could flag a given website for further investigation. This could serve as an early warning system for blocklist providers and other detection mechanisms.

The third set of applications would be those deployed within an organization as opposed to those used to protect everyday internet users more broadly. Private companies and government agencies—particularly those in the defense, energy, and financial sectors—place a much higher premium on security than the average user. The cost of a single compromise is often much higher

for these types of organizations. As a result, they are more willing to accept a higher false positive rate if that means there is a lower chance of a breach. Therefore, an intervention system as described in the first use case could be well suited for these environments, even if it is insufficient for general use. In addition, deployment of this technology could also be used to assess vulnerabilities within an organization. Evaluating the browsing behavior of employees could be used to identify areas of risk, allocate security resources accordingly, and target additional security training where required.

Lastly, the model proposed in this paper was trained on the browsing activities and the exposures of *all* users. On the one hand, in doing so, it can provide better coverage to identify different types of exposure. On the other hand, it is not tailored to users with different risk tolerance levels. For example, a cautious user may want to be notified when a page is considered malicious with respect to all kinds of threats, while another user may be well-trained to identify phishing scams and may want to be notified only when the page serves, e.g., a drive-by download. To achieve such a finer level of granularity, an organization could fine-tune the models with specific types of exposures based on users' preferences.

Performance aside, other factors may influence the deployment decision. For example, in-browser deployment might be more desirable from a privacy standpoint, and network-level deployment might be more desirable if reducing local computation cost is a concern.

5.4 Future Work

To bring this technology closer to deployment, several technical improvements should be evaluated. This includes refinement of the core features necessary to identify exposure. In particular, the category of a website, which was one of the most important features in our models, could be greatly improved upon. The accuracy of these categories is one opportunity for improvement, as well as the way they are modeled. For example, instead of single labels, using multi-label categories may be a more robust approach. In addition, further study of performance across a variety of devices, languages, and cultures is needed. Future work should investigate how tailoring these models to specific environments, risk preferences, organizations, and individuals affects performance.

While browser warnings and interventions have been well-studied, the proactive nature of our models creates new possibilities. Examples include providing real-time feedback about the risk to users as they browse, or silently throttling a connection when a user enters a riskier territory. It is not clear how effective these interventions might be and how they might compare to traditional warnings.

There are also opportunities that extend beyond exposure to malicious content where behavior-based models may be helpful. It may be possible to identify when users are exposed to other online harms like violations of privacy and misinformation. In addition, understanding what "normal" patterns of user browsing behavior look like could enable better detection of bots and other non-user web traffic.

Despite the effectiveness of the sequential representation of user behaviors and the additional features, we believe the performance could be further improved with a more expressive model, such as a transformer. We leave the implementation of such a model to future work.

Finally, numerous policy questions should be considered. For example, if behavior-based security systems were deployed at scale, what would be the cost of implementation? How would network congestion and browsing speeds be affected? What are the costs of false positives, and are there websites or users that would be disproportionately affected? Would additional regulation be required to mitigate or prevent adversarial uses of this technology? These considerations, often investigated only after a technology is deployed, are vital to address proactively and require additional interdisciplinary research.

6 Conclusion

Behavior-based modeling for detecting exposure was proposed to mitigate the gap between malicious content creation and detection that blocklist-based mechanisms suffer from. However, limitations to the effectiveness and the robustness of the current state-of-the-art models hinder their use in helping protect users online. This study represents an important step in bringing behavior-based security mechanisms closer to a deployable solution by developing a model that greatly outperforms that of previous work and by shedding light on how certain behavioral factors contribute to exposure.

Our proposed model leverages a time-series representation of user browse behavior to capture temporal ordering that has not been employed for behavior-based detection models before. Building on this representation, we introduced additional features that, together with the time-series representation, greatly improve performance compared to previous work. This includes a 93% to 145% increase in F1-score and a 15% to 18% improvement in AUC. In addition, we demonstrate that the proposed models are robust over time and across datasets of disparate users. We assess the importance of different features and the factors behind incorrect classifications to aid system explainability. We then provide a detailed discussion of the limitations of behavior-based models, the corresponding risks for their use, potential applications in security, and opportunities for future work.

Acknowledgments

This work was partially supported by the National Security Agency (NSA) Science of Security Lablet at Carnegie Mellon University (H9823014C0140), by the Defence Science and Technology Agency (CNZ000832), and through a Carnegie Bosch Institute (CBI) Fellowship.

References

[1] Sahar Abdelnabi, Katharina Krombholz, and Mario Fritz. 2020. VisualPhishNet: Zero-Day Phishing Website Detection by Visual Similarity. In *Proceedings of the 2020 ACM SIGSAC Conference on Computer and Communications Security* (Virtual Event, USA) *(CCS '20)*. Association for Computing Machinery, New York, NY, USA, 1681–1698. https://doi.org/10.1145/3372297.3417233

[2] Eihal Alowaisheq, Peng Wang, Sumayah Alrwais, Xiaojing Liao, XiaoFeng Wang, Tasneem Alowaisheq, Xianghang Mi, Siyuan Tang, and Baojun Liu. 2019. Cracking the Wall of Confinement: Understanding and Analyzing Malicious Domain Take-downs. In *Proceedings of the Network and Distributed System Security Symposium (NDSS)*. Internet Society, San Diego, CA, USA. https://doi.org/10.14722/ndss.2019.23243

[3] Abdullah Alshehri, Nayeem Khan, Ali Alowayr, and Mohammed Yahya Alghamdi. 2023. Cyberattack Detection Framework Using Machine Learning and User Behavior Analytics. *Computer Systems Science and Engineering* 44, 2 (2023), 1679–1689. https://doi.org/10.32604/csse.2023.026526

[4] The EasyList authors (https://easylist.to/). 2025. EasyList. https://easylist.to/. Accessed: 2025-10-09.

[5] Simon Bell and Peter Komisarczuk. 2020. An Analysis of Phishing Blacklists: Google Safe Browsing, OpenPhish, and PhishTank. In *Proceedings of the Australasian Computer Science Week Multiconference* (Melbourne, VIC, Australia) *(ACSW '20)*. Association for Computing Machinery, New York, NY, USA, Article 3, 11 pages. https://doi.org/10.1145/3373017.3373020

[6] Fabrício Benevenuto, Tiago Rodrigues, Meeyoung Cha, and Virgílio Almeida. 2009. Characterizing User Behavior in Online Social Networks. In *Proceedings of the 9th ACM SIGCOMM Conference on Internet Measurement* (Chicago, Illinois, USA) *(IMC '09)*. ACM, New York, NY, USA, 49–62. https://doi.org/10.1145/1644893.1644900

[7] Sruti Bhagavatula, Lujo Bauer, and Apu Kapadia. 2021. What breach? Measuring online awareness of security incidents by studying real-world browsing behavior. In *Proceedings of the 2021 European Symposium on Usable Security*. 180–199.

[8] Leyla Bilge, Yufei Han, and Matteo Dell'Amico. 2017. RiskTeller: Predicting the Risk of Cyber Incidents. In *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security* (Dallas, Texas, USA) *(CCS '17)*. Association for Computing Machinery, New York, NY, USA, 1299–1311. https://doi.org/10.1145/3133956.3134022

[9] United States Census Bureau. 2017. 2017: ACS 5-Year Estimates Data Profiles. https://data.census.gov/table/ACSDP5Y2017.DP05. Accessed: 2025-10-09.

[10] United States Census Bureau. 2019. Educational Attainment in the United States: 2018. https://www.census.gov/data/tables/2018/demo/education-attainment/cps-detailed-tables.html. Accessed: 2025-10-09.

[11] Davide Canali, Leyla Bilge, and Davide Balzarotti. 2014. On the Effectiveness of Risk Prediction Based on Users Browsing Behavior. In *Proceedings of the 9th ACM Symposium on Information, Computer and Communications Security* (Kyoto, Japan) *(ASIA CCS '14)*. ACM, New York, NY, USA, 171–182. https://doi.org/10.1145/2590296.2590347

[12] Lara D. Catledge and James E. Pitkow. 1995. Characterizing browsing strategies in the World-Wide web. *Computer Networks and ISDN Systems* 27, 6 (1995), 1065 – 1073. https://doi.org/10.1016/0169-7552(95)00043-7 Proceedings of the Third International World-Wide Web Conference.

[13] Sidharth Chhabra, Anupama Aggarwal, Fabricio Benevenuto, and Ponnurangam Kumaraguru. 2011. Phi.Sh/$oCiaL: The Phishing Landscape through Short URLs. In *Proceedings of the 8th Annual Collaboration, Electronic Messaging, Anti-Abuse and Spam Conference* (Perth, Australia) *(CEAS '11)*. Association for Computing Machinery, New York, NY, USA, 92–101. https://doi.org/10.1145/2030376.2030387

[14] Kyle Crichton, Nicolas Christin, and Lorrie Faith Cranor. 2021. How Do Home Computer Users Browse the Web? *ACM Trans. Web* 16, 1, Article 3 (Sept. 2021), 27 pages. https://doi.org/10.1145/3473343

[15] Louis F. DeKoven, Audrey Randall, Ariana Mirian, Gautam Akiwate, Ansel Blume, Lawrence K. Saul, Aaron Schulman, Geoffrey M. Voelker, and Stefan Savage. 2019. Measuring Security Practices and How They Impact Security. In *Proceedings of the Internet Measurement Conference* (Amsterdam, Netherlands) *(IMC '19)*. Association for Computing Machinery, New York, NY, USA, 36–49. https://doi.org/10.1145/3355369.3355571

[16] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 conference of the North American chapter of the association for computational linguistics: human language technologies, volume 1 (long and short papers)*. 4171–4186.

[17] DomCop. 2025. Open Page Rank. https://www.domcop.com/openpagerank. Accessed: 2025-10-09.

[18] Alain Forget, Saranga Komanduri, Alessandro Acquisti, Nicolas Christin, Lorrie Cranor, and Rahul Telang. 2014. Security Behavior Observatory: Infrastructure for Long-term Monitoring of Client Machines (CMU-CyLab-14-009). *CyLab* 1, 1 (Jul 2014), 1. https://doi.org/10.1184/R1/6468056.v1

[19] Ian J. Goodfellow, Jonathon Shlens, and Christian Szegedy. 2015. Explaining and Harnessing Adversarial Examples. arXiv:1412.6572 [stat.ML] https://arxiv.org/abs/1412.6572

[20] Google. 2025. Google Safe Browsing. https://safebrowsing.google.com/. Accessed: 2025-10-09.

[21] Guofei Gu, Roberto Perdisci, Junjie Zhang, and Wenke Lee. 2008. BotMiner: Clustering Analysis of Network Traffic for Protocol- and Structure-Independent Botnet Detection. In *Proceedings of the 17th Conference on Security Symposium* (San Jose, CA) *(SS'08)*. USENIX Association, USA, 139–154.

[22] Guofei Gu, Junjie Zhang, and Wenke Lee. 2008. BotSniffer: Detecting Botnet Command and Control Channels in Network Traffic. In *Proceedings of the Network and Distributed System Security Symposium (NDSS)*. Internet Society.

[23] Hana Habib, Jessica Colnago, Vidya Gopalakrishnan, Sarah Pearman, Jeremy Thomas, Alessandro Acquisti, Nicolas Christin, and Lorrie Faith Cranor. 2018. Away from Prying Eyes: Analyzing Usage and Understanding of Private Browsing. In *Proceedings of the Fourteenth USENIX Conference on Usable Privacy and Security* (Baltimore, MD, USA) *(SOUPS '18)*. USENIX Association, USA, 159–175.

[24] Xiao Han, Nizar Kheir, and Davide Balzarotti. 2016. PhishEye: Live Monitoring of Sandboxed Phishing Kits. In *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security* (Vienna, Austria) *(CCS '16)*. Association for Computing Machinery, New York, NY, USA, 1402–1413. https://doi.org/10.1145/2976749.2978330

[25] Jian Hu, Hua-Jun Zeng, Hua Li, Cheng Niu, and Zheng Chen. 2007. Demographic prediction based on user's browsing behavior. *16th International World Wide Web Conference, WWW2007*, 151–160. https://doi.org/10.1145/1242572.1242594

[26] Luca Invernizzi, Stanislav Miskovic, Ruben Torres, Christopher Kruegel, Sabyasachi Saha, Giovanni Vigna, Sung-Ju Lee, and Marco Mellia. 2014. Nazca: Detecting Malware Distribution in Large-Scale Networks. In *Proceedings of the Network and Distributed System Security Symposium (NDSS)*. Internet Society.

[27] Luca Invernizzi, Kurt Thomas, Alexandros Kapravelos, Oxana Comanescu, Jean-Michel Picod, and Elie Bursztein. 2016. Cloak of Visibility: Detecting When Machines Browse a Different Web. In *2016 IEEE Symposium on Security and Privacy (SP)*. 743–758. https://doi.org/10.1109/SP.2016.50

[28] Naurin Farooq Khan, Naveed Ikram, Sumera Saleem, and Saad Zafar. 2023. Cyber-security and risky behaviors in a developing country context: a Pakistani perspective. *Security Journal* 36, 2 (2023), 373–405. https://doi.org/10.1057/s41284-022-00343-4

[29] MASAHARU BAN KOSUKE TOSHI. 2022. Internet Explorer shutdown to cause Japan headaches 'for months'. Nikkei Asia. https://asia.nikkei.com/Business/Technology/Internet-Explorer-shutdown-to-cause-Japan-headaches-for-months.

[30] Ravi Kumar and Andrew Tomkins. 2010. A Characterization of Online Browsing Behavior. In *Proceedings of the 19th International Conference on World Wide Web* (Raleigh, North Carolina, USA) *(WWW '10)*. Association for Computing Machinery, New York, NY, USA, 561–570. https://doi.org/10.1145/1772690.1772748

[31] Udesh Kumarasinghe, Mohamed Nabeel, and Charitha Elvitigala. 2024. Blocklist-Forecast: Proactive Domain Blocklisting by Identifying Malicious Hosting Infrastructure. In *Proceedings of the 27th International Symposium on Research in Attacks, Intrusions and Defenses*. 35–48.

[32] Janette Lehmann, Mounia Lalmas, Georges Dupret, and Ricardo Baeza-Yates. 2013. Online Multitasking and User Engagement. In *Proceedings of the 22Nd ACM International Conference on Information & Knowledge Management* (San Francisco, California, USA) *(CIKM '13)*. ACM, New York, NY, USA, 519–528. https://doi.org/10.1145/2505515.2505543

[33] Fanny Lalonde Lévesque, Sonia Chiasson, Anil Somayaji, and José M. Fernandez. 2018. Technological and Human Factors of Malware Attacks: A Computer Security Clinical Trial Approach. *ACM Trans. Priv. Secur.* 21, 4, Article 18 (July 2018), 30 pages. https://doi.org/10.1145/3210311

[34] Marc Liberatore and Brian Neil Levine. 2006. Inferring the Source of Encrypted HTTP Connections. In *Proceedings of the 13th ACM Conference on Computer and Communications Security* (Alexandria, Virginia, USA) *(CCS '06)*. Association for Computing Machinery, New York, NY, USA, 255–263. https://doi.org/10.1145/1180405.1180437

[35] Chao Liu, Ryen W. White, and Susan Dumais. 2010. Understanding Web Browsing Behaviors through Weibull Analysis of Dwell Time. In *Proceedings of the 33rd International ACM SIGIR Conference on Research and Development in Information Retrieval* (Geneva, Switzerland) *(SIGIR '10)*. Association for Computing Machinery, New York, NY, USA, 379–386. https://doi.org/10.1145/1835449.1835513

[36] Yang Liu, Armin Sarabi, Jing Zhang, Parinaz Naghizadeh, Manish Karir, Michael Bailey, and Mingyan Liu. 2015. Cloudy with a Chance of Breach: Forecasting Cyber Security Incidents. In *24th USENIX Security Symposium (USENIX Security 15)*. USENIX Association, Washington, D.C., 1009–1024. https://www.usenix.org/conference/usenixsecurity15/technical-sessions/presentation/liu

[37] Christian Ludl, Sean McAllister, Engin Kirda, and Christopher Kruegel. 2007. On the Effectiveness of Techniques to Detect Phishing Sites. In *Detection of Intrusions and Malware, and Vulnerability Assessment*, Bernhard M. Hämmerli and Robin Sommer (Eds.). Springer Berlin Heidelberg, Berlin, Heidelberg, 20–39.

[38] Siddharth M Madikeri and Vijay K Madisetti. 2024. Ad Blockers & Online Privacy: A Comparative Analysis of Privacy Enhancing Technologies (PET). *Journal of Software Engineering and Applications* 17, 5 (2024), 378–395.

[39] B. McKenzie and A. Cockburn. 2001. An empirical analysis of web page revisitation. In *Proceedings of the 34th Annual Hawaii International Conference on System Sciences*. 9 pp.–.

[40] Marcello Meschini, Giorgio Di Tizio, Marco Balduzzi, and Fabio Massacci. 2024. A Case-Control Study to Measure Behavioral Risks of Malware Encounters in Organizations. *IEEE Transactions on Information Forensics and Security* 19 (2024), 9419–9432. https://doi.org/10.1109/TIFS.2024.3456960

[41] Leigh Metcalf and Jonathan M. Spring. 2015. Blacklist Ecosystem Analysis: Spanning Jan 2012 to Jun 2014. In *Proceedings of the 2nd ACM Workshop on Information Sharing and Collaborative Security* (Denver, Colorado, USA) *(WISCS '15)*. Association for Computing Machinery, New York, NY, USA, 13–22. https://doi.org/10.1145/2808128.2808129

[42] Mohamed Nabeel, Issa M. Khalil, Bei Guan, and Ting Yu. 2020. Following Passive DNS Traces to Detect Stealthy Malicious Domains Via Graph Inference. *ACM Trans. Priv. Secur.* 23, 4, Article 17 (July 2020), 36 pages. https://doi.org/10.1145/3401897

[43] Terry Nelms, Roberto Perdisci, and Mustaque Ahamad. 2013. ExecScent: Mining for New C&C Domains in Live Networks with Adaptive Control Protocol Templates. In *22nd USENIX Security Symposium (USENIX Security 13)*. USENIX Association, Washington, D.C., 589–604. https://www.usenix.org/conference/usenixsecurity13/technical-sessions/paper/nelms

[44] Terry Nelms, Roberto Perdisci, Manos Antonakakis, and Mustaque Ahamad. 2015. WebWitness: Investigating, Categorizing, and Mitigating Malware Download Paths. In *24th USENIX Security Symposium (USENIX Security 15)*. USENIX Association, Washington, D.C., 1025–1040. https://www.usenix.org/conference/usenixsecurity15/technical-sessions/presentation/nelms

[45] Philip J. Nero, Brad Wardman, Heith Copes, and Gary Warner. 2011. Phishing: Crime that pays. In *2011 eCrime Researchers Summit*. 1–10. https://doi.org/10.1109/eCrime.2011.6151979

[46] Dong Nie, Yue Ning, and Tingshao Zhu. 2012. Predicting Mental Health Status in the Context of Web Browsing. In *2012 IEEE/WIC/ACM International Conferences on Web Intelligence and Intelligent Agent Technology*, Vol. 3. 185–189. https://doi.org/10.1109/WI-IAT.2012.196

[47] Hartmut Obendorf, Harald Weinreich, Eelco Herder, and Matthias Mayer. 2007. Web Page Revisitation Revisited: Implications of a Long-Term Click-Stream Study of Browser Usage. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems* (San Jose, California, USA) *(CHI '07)*. Association for Computing Machinery, New York, NY, USA, 597–606. https://doi.org/10.1145/1240624.1240719

[48] Adam Oest, Yeganeh Safaei, Adam Doupé, Gail-Joon Ahn, Brad Wardman, and Kevin Tyers. 2019. PhishFarm: A Scalable Framework for Measuring the Effectiveness of Evasion Techniques against Browser Phishing Blacklists. In *2019 IEEE Symposium on Security and Privacy (SP)*. 1344–1361. https://doi.org/10.1109/SP.2019.00049

[49] Adam Oest, Penghui Zhang, Brad Wardman, Eric Nunes, Jakub Burgis, Ali Zand, Kurt Thomas, Adam Doupé, and Gail-Joon Ahn. 2020. Sunrise to Sunset: Analyzing the End-to-end Life Cycle and Effectiveness of Phishing Attacks at Scale. In *29th USENIX Security Symposium (USENIX Security 20)*. USENIX Association, 361–377. https://www.usenix.org/conference/usenixsecurity20/presentation/oest-sunrise

[50] U.S. Bureau of Labor Statistics. 2025. Employment Projections. https://www.bls.gov/emp/tables/occupational-projections-and-characteristics.htm. Accessed: 2025-10-09.

[51] Michael Ovelgönne, Tudor Dumitraş, B. Aditya Prakash, V. S. Subrahmanian, and Benjamin Wang. 2017. Understanding the Relationship between Human Behavior and Susceptibility to Cyber Attacks: A Data-Driven Approach. *ACM Trans. Intell. Syst. Technol.* 8, 4, Article 51 (March 2017), 25 pages. https://doi.org/10.1145/2890509

[52] M. Zubair Rafique, Tom van Goethem, Wouter Joosen, Christophe Huygens, and Nick Nikiforakis. 2016. It's Free for a Reason: Exploring the Ecosystem of Free Live Streaming Services. In *NDSS*.

[53] Arun Raghuramu, Parth H. Pathak, Hui Zang, Jinyoung Han, Chang Liu, and Chen-Nee Chuah. 2016. Uncovering the footprints of malicious traffic in wireless/mobile networks. *Computer Communications* 95 (2016), 95–107. https://doi.org/10.1016/j.comcom.2016.04.011 Mobile Traffic Analytics.

[54] Moheeb Abu Rajab, Lucas Ballard, Noe Lutz, Panayiotis Mavrommatis, and Niels Provos. 2013. CAMP: Content-Agnostic Malware Protection. In *Network and Distributed Systems Security Symposium (NDSS)*. USA. http://cs.jhu.edu/~moheeb/aburajab-ndss-13.pdf

[55] Rohit Ranjan and Shashi Shekhar Kumar. 2022. User behaviour analysis using data analytics and machine learning to predict malicious user versus legitimate user. *High-confidence computing* 2, 1 (2022), 100034.

[56] Konrad Rieck, Tammo Krueger, and Andreas Dewald. 2010. Cujo: Efficient Detection and Prevention of Drive-by-Download Attacks. In *Proceedings of the 26th Annual Computer Security Applications Conference* (Austin, Texas, USA) *(ACSAC '10)*. Association for Computing Machinery, New York, NY, USA, 31–39. https://doi.org/10.1145/1920261.1920267

[57] Takaya Saito and Marc Rehmsmeier. 2015. The precision-recall plot is more informative than the ROC plot when evaluating binary classifiers on imbalanced datasets. *PloS one* 10, 3 (2015), e0118432.

[58] Mahmood Sharif, Jumpei Urakawa, Nicolas Christin, Ayumu Kubota, and Akira Yamada. 2018. Predicting Impending Exposure to Malicious Content from User Behavior. In *Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security* (Toronto, Canada) *(CCS '18)*. Association for Computing Machinery, New York, NY, USA, 1487–1501. https://doi.org/10.1145/3243734.3243779

[59] Yun Shen, Enrico Mariconti, Pierre Antoine Vervier, and Gianluca Stringhini. 2018. Tiresias: Predicting Security Events Through Deep Learning. In *Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security* (Toronto, Canada) *(CCS '18)*. Association for Computing Machinery, New York, NY, USA, 592–605. https://doi.org/10.1145/3243734.3243811

[60] Steve Sheng, Brad Wardman, Gary Warner, Lorrie Faith Cranor, Jason I. Hong, and Chengshan Zhang. 2009. An Empirical Analysis of Phishing Blacklists. In *CEAS 2009*.

[61] Kyle Soska and Nicolas Christin. 2014. Automatically Detecting Vulnerable Websites Before They Turn Malicious. In *23rd USENIX Security Symposium (USENIX Security 14)*. USENIX Association, San Diego, CA, 625–640. https://www.usenix.org/conference/usenixsecurity14/technical-sessions/presentation/soska

[62] Qixiang Sun, D.R. Simon, Yi-Min Wang, W. Russell, V.N. Padmanabhan, and Lili Qiu. 2002. Statistical identification of encrypted Web browsing traffic. In *Proceedings 2002 IEEE Symposium on Security and Privacy*. 19–30. https://doi.org/10.1109/SECPRI.2002.1004359

[63] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. 2015. Going deeper with convolutions. In *Proceedings of the IEEE conference on computer vision and pattern recognition*. 1–9.

[64] Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jon Shlens, and Zbigniew Wojna. 2016. Rethinking the inception architecture for computer vision. In *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2818–2826.

[65] Linda Tauscher and Saul Greenberg. 1997. Revisitation Patterns in World Wide Web Navigation. In *Proceedings of the ACM SIGCHI Conference on Human Factors in Computing Systems* (Atlanta, Georgia, USA) *(CHI '97)*. Association for Computing Machinery, New York, NY, USA, 399–406. https://doi.org/10.1145/258549.258816

[66] Phani Vadrevu, Babak Rahbarinia, Roberto Perdisci, Kang Li, and Manos Antonakakis. 2013. Measuring and Detecting Malware Downloads in Live Network Traffic. In *Computer Security – ESORICS 2013*, Jason Crampton, Sushil Jajodia, and Keith Mayes (Eds.). Springer Berlin Heidelberg, Berlin, Heidelberg, 556–573.

[67] Gang Wang, Tristan Konolige, Christo Wilson, Xiao Wang, Haitao Zheng, and Ben Y. Zhao. 2013. You Are How You Click: Clickstream Analysis for Sybil Detection. In *22nd USENIX Security Symposium (USENIX Security 13)*. USENIX Association, Washington, D.C., 241–256. https://www.usenix.org/conference/usenixsecurity13/technical-sessions/presentation/wang

[68] Gang Wang, Xinyi Zhang, Shiliang Tang, Christo Wilson, Haitao Zheng, and Ben Y. Zhao. 2017. Clickstream User Behavior Models. *ACM Trans. Web* 11, 4, Article 21 (July 2017), 37 pages. https://doi.org/10.1145/3068332

[69] Miranda Wei, Jaron Mink, Yael Eiger, Tadayoshi Kohno, Elissa M Redmiles, and Franziska Roesner. 2024. {SoK}(or {SoLK?)}: On the Quantitative Study of Sociodemographic Factors and Computer Security Behaviors. In *33rd USENIX Security Symposium (USENIX Security 24)*. 7011–7030.

[70] Harald Weinreich, Hartmut Obendorf, Eelco Herder, and Matthias Mayer. 2008. Not Quite the Average: An Empirical Study of Web Use. *ACM Trans. Web* 2, 1, Article 5 (mar 2008), 31 pages. https://doi.org/10.1145/1326561.1326566

[71] Senhao Wen, Zhiyuan Zhao, and Hanbing Yan. 2018. Detecting Malicious Websites in Depth through Analyzing Topics and Web-Pages. In *Proceedings of the 2nd International Conference on Cryptography, Security and Privacy* (Guiyang, China) *(ICCSP 2018)*. Association for Computing Machinery, New York, NY, USA, 128–133. https://doi.org/10.1145/3199478.3199500

[72] Gilbert Wondracek, Thorsten Holz, Christian Platzer, Engin Kirda, and Christopher Kruegel. 2010. Is the Internet for Porn? An Insight into the Online Adult Industry. (June 2010).

[73] Songyun Wu, Bo Wang, Zhiliang Wang, Shuhan Fan, Jiahai Yang, and Jia Li. 2022. Joint prediction on security event and time interval through deep learning. *Computers & Security* 117 (2022), 102696. https://doi.org/10.1016/j.cose.2022.102696

[74] Li Xu, Zhenxin Zhan, Shouhuai Xu, and Keying Ye. 2013. Cross-Layer Detection of Malicious Websites. In *Proceedings of the Third ACM Conference on Data and Application Security and Privacy* (San Antonio, Texas, USA) *(CODASPY '13)*. Association for Computing Machinery, New York, NY, USA, 141–152. https://doi.org/10.1145/2435349.2435366

[75] Akira Yamada, Kyle Crichton, Yukiko Sawaya, Jin-Dong Dong, Sarah Pearman, Ayumu Kubota, and Nicolas Christin. 2022. On recruiting and retaining users for security-sensitive longitudinal measurement panels. In *Eighteenth Symposium on Usable Privacy and Security (SOUPS 2022)*. USENIX Association, Boston, MA, 347–366. https://www.usenix.org/conference/soups2022/presentation/yamada

[76] Haimo Zhang and Shengdong Zhao. 2011. Measuring Web Page Revisitation in Tabbed Browsing. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems* (Vancouver, BC, Canada) *(CHI '11)*. Association for Computing Machinery, New York, NY, USA, 1831–1834. https://doi.org/10.1145/1978942.1979207

[77] Junjie Zhang, Christian Seifert, Jack W. Stokes, and Wenke Lee. 2011. ARROW: GenerAting SignatuRes to Detect DRive-By DOWnloads. In *Proceedings of the 20th International Conference on World Wide Web* (Hyderabad, India) *(WWW '11)*. Association for Computing Machinery, New York, NY, USA, 187–196. https://doi.org/10.1145/1963405.1963435

[78] Yue Zhang, Serge Egelman, Lorrie Cranor, and Jason Hong. 2005. Phinding Phish: Evaluating Anti-Phishing Tools. https://doi.org/10.1184/R1/6470321.v1

[79] Yue Zhang, Jason I. Hong, and Lorrie F. Cranor. 2007. Cantina: A Content-Based Approach to Detecting Phishing Web Sites. In *Proceedings of the 16th International Conference on World Wide Web* (Banff, Alberta, Canada) *(WWW '07)*. Association for Computing Machinery, New York, NY, USA, 639–648. https://doi.org/10.1145/1242572.1242659

[80] Benjamin Zi Hao Zhao, Muhammad Ikram, Hassan Jameel Asghar, Mohamed Ali Kaafar, Abdelberi Chaabane, and Kanchana Thilakarathna. 2019. A Decade of Mal-Activity Reporting: A Retrospective Analysis of Internet Malicious Activity Blacklists. In *Proceedings of the 2019 ACM Asia Conference on Computer and Communications Security* (Auckland, New Zealand) *(Asia CCS '19)*. Association for Computing Machinery, New York, NY, USA, 193–205. https://doi.org/10.1145/3321705.3329834

A Additional LSTM Features

Additional LSTM Features

New Site in Session: A indicator of whether the user has visited this site earlier in the same browsing session. If the user has returned to a given website more than once in the course of a browsing session then it is unlikely to contain malicious content.
Year: Since attacks, and browsing patterns to a lesser extent, change over time and the SBO dataset covers four years of browsing, we capture the year that the browsing occurred using binary flags.
Event Type: In the SBO dataset, we have access to a range of user behaviors beyond web requests. Data available includes detailed browsing information like the type of web-request triggering event (e.g. link, bookmark, reload, subframe, etc.) or the creation, activation, or closing of a browser tab. In addition, non-browsing events include user interactions with a non-browser application, logging on and off, and powering the machine on and off.
Private Browsing: Previous work has found that while private browsing does help protect certain aspects of privacy, a common misconception is that it also affords additional security protections [23].

We hypothesize that users may enable private browsing when they know they engaging in riskier browsing. This is indicated by a binary feature in the model.
Current Browser Tab: We hypothesize that patterns in how users employ multiple browser tabs, and how websites automatically open up new tabs to display additional web page content can play a role in exposure. For example, many free streaming websites use pop-ups which force a user to engage with the potentially malicious content before allowing them to watch a video [52]. This is captured using a vector of 50 binary flags, the maximum number of different tabs that could be employed in a 50 event sequence, that represent which tab is actively being used.
Tabs and Windows Open: The total number of browser tabs and windows currently open on the user's machine at each point during the browsing session. Rapid increases in the number of open tabs and windows may indicate anomalous behavior such as a site automatically spawning additional web pages.
Active Browsing Depth and Time: The number of seconds and events since the user last switched to their browser window.
Tab Browsing Depth and Time: The number of seconds and events since the user last switched to the currently use browser tab.
Navigation Qualifiers: These four variables indicate whether the user's navigation was initiated by 1) a user's interaction with the address bar, 2) the forward or back buttons, 3) JavaScript or meta refresh tags on a web page, or 4) redirects caused by HTTP headers sent from the server. This is of particular interest to capture navigations that were triggered automatically by the server or embedded code which could link the user to a malicious page without their direct action.
Application Category: Visits to malicious websites can also be triggered from outside of the browser. Two common examples are social engineering attacks received through an email or instant messaging tool and malicious code embedded in downloaded software. To capture these attack vectors, the 53 application categories, summarized in Appendix C, are each represented by a binary flag.

B Website Categories

Each website in the SBO and the ST datasets were assigned a category (see Section 3.3.1 for detail). Table 4 summarizes the 66-category and 49-category taxonomy used in the SBO and the ST datasets respectively, along with the mapping between the two.

C Application Categories

The application categories used in the SBO LSTM-Browser model are shown in Table 5 . These categories were based on those used in previous work [14].

Table 4. Website categories across datasets.

Mapping	SBO	ST
Art	Performing Arts	Art
	Visual Arts	Photography
Business	Business	Agriculture
	Crowdworking	Business
	Job Search	
	Productivity	
Computers	CDNs	Computers
	Computers	Software
	Electronics	Web Development
	Open Source Content	
	Programming	
Cultural	Cultural	Cultural
Education	Education	Education
		Universities
Financial	Financial	Financial
Food & Drink	Food & Drink	Food
	Entertainment	Restaurants & Bars
Gaming	Gambling	Gaming
	Gaming	
Government	Government	Government
		Local
		Politics
Health	Alternative Health	Dentistry
	Healthcare	Healthcare
	Mental Health	Medicine
	Reproductive Health	
	Senior Health	
Home & Garden	Family	Home & Garden
	Home & Garden	
Industrial	Industrial	Construction
		Industrial
Legal	Legal	Legal
Leisure	Hobbies	Leisure
	Leisure	
Literature	Anime & Comics	Literature
	Literature	
Marketing & Ads	Advertising	Marketing & Ads
	Ad Redirects	
Movies & TV	Movies & TV	Movies & TV
	Streaming	Entertainment
	Celebrity	
Music	Music	Music
News & Media	News & Media	News & Media
Organizations	Organizations	Organizations
	Philanthropy	Clubs
Outdoors	Outdoors	Outdoors
Personal Pages	Personal Pages	People
		Personal Pages
Pets	Pets	Pets
Real Estate	Real Estate	Real Estate
Religion	Religion	Religion
Science	Alternative Science	Science
	Humanities	
	Science	
	Social Sciences	
Services	Classifieds	Services
	Services	
Shopping	Shopping	Clothing
	Rewards & Freebies	Shopping
Internet	Adult	Internet
	Communication	
	Email	
	Online Entertainment	
	Reference	
	Search	
	Social	
	Web Portal	
Society	Genealogy	Society
	LGBTQ+	
	Dating	
	Social Issues	
	Subcultures	
Sports	Sports	Sports
Travel	Travel	Accommodation
		Hotels
		Travel
Unknown	Unknown	
Vehicles	Vehicles	Vehicles

Table 5. Hierarchical category structure for labeling applications that participant's used on their computer.

Category	Types of Applications
Adult	Pornographic videos and games
Animation and Comics	Manga readers, animation tools
Audio Editing	Audio and music editing
Business	Billing, payroll, accounting tools
Computers and Programming	Program editors, IDEs, debuggers
Communication	Email, chat clients, video conferencing, e-cards
Connected Devices	Printers, scanners, USB devices
File Sharing	Torrents, download managers
File System and Storage	File explorer, cloud storage, file transfer
Finances	Investment tools, cryptocurrency wallets
Food and Drink	Recipes, cooking instruction
Hobbies	Drone software, amateur radio, music tools
Home and Garden	Home design, 3D designers
Gambling	Casino applications, betting tools
Gaming	Video games, streaming software, mod editors
Image Editing	Photo viewers, photo editors, graphic design
Install/Uninstall	Installing and uninstalling programs
Lifestyle	Fitness, calorie trackers, wearables
Literature	E-Readers
Media Player	Adobe Flash, video players
Navigation	Start menu, search
News and Media	News articles, weather apps
Notifications/Popups	Notification windows
Office Tools	Text, word, spreadsheet, and presentation editors
Productivity	Productivity utilities, data analysis software
Reference	Maps, dictionary and thesaurus
Remote Connectivity	Remote desktop, SSH tools
Science	3D Modeling, scientific imaging and sensors
Security	Antivirus, malware removal tools
Shopping	Retail portals, discount tools, app stores
Social	Social media desktop applications
Social Science	Decision and simulation tools
Streaming	Netflix, Hulu, Spotify desktop applications
System Configuration	Display, audio, power, and network settings
System Repair	Data backup, data recovery, system recovery
Unknown	Typically unreadable sensor information
Video Editing	Video editing, media converting or burning
Web Browsing	Chrome, Firefox, Internet Explorer, Opera, Tor