AJNA: Anti-Phishing JS-based Visual Analysis, to Mitigate Users' Excessive Trust in SSL/TLS

Pernelle Mensah*†, Gregory Blanc*, Kazuya Okada†, Daisuke Miyamoto‡ and Youki Kadobayashi†

*Institut Mines-Télécom/Télécom SudParis
Université Paris-Saclay
9 rue Charles Fourier, 91011 Évry, France
pernelle.mensah@gmail.com, gregory.blanc@telecom-sudparis.eu

†Graduate School of Information Science
Nara Institute of Science and Technology
8916-5 Takayama, Ikoma, Nara, 630-0192 Japan
kazuya-o@is.naist.jp, youki-k@is.aist-nara.ac.jp

‡Information Technology Center
The University of Tokyo
2-11-16 Yayoi, Bunkyo-ku, Tokyo, 113-8658 Japan
daisu-mi@nc.u-tokyo.ac.jp

***Abstract*—HTTPS websites are often considered safe by the users, due to the use of the SSL/TLS protocol. As a consequence phishing web pages delivered via this protocol benefit from that higher level of trust as well.**
In this paper, we assessed the relevance of heuristics such as the certificate information, the SSL/TLS protocol version and cipher-suite chosen by the servers, in the identification of phishing websites. We concluded that they were not discriminant enough, due to the close profiles of phishing and legitimate sites. Moreover, considering phishing pages hosted on cloud service platform or hacked domains, we identified that the users could easily be fooled by the certificate presented, since it would belong to the rightful owner of the website.
Hence, we further examined HTTPS phishing websites hosted on hacked domains, in order to propose a detection method based on their visual identities. Indeed, the presence of a parasitic page on a domain is a disruption to the overall visual coherence of the original site. By designing an intelligent perception system responsible for extracting and comparing these divergent renderings, we were able to spot phishing pages with an accuracy of 87% to 92%.

I. Introduction

In our modern age, more and more professional, commercial, and banking exchanges happen online every day. This dematerialization of communication opens the gates to criminal behaviours such as phishing, which is the action of impersonating a legitimate website, in order to deceive the user and incite him to reveal personal details such as login credentials or credit card information. The phishing activity represents quite a lucrative business for cybercriminals. Indeed, it has generated close to 8.5 billion dollars of losses in the US for the year 2014, according to the Internet Crime Complaint Center (IC3) of the Federal Bureau of Investigation (FBI) [1].

Regarding this threat, various measures have been taken such as the development of technical solutions and the securing of websites by leveraging the Secure Sockets Layer (SSL) protocol whose main goal is the authentication of the server, as well as the privacy and integrity of the communication. In other words, this protocol is supposed to guarantee that the user is really exchanging information with the website he intended to join, and that no external party can have knowledge of this data or alter it. A communication over SSL is mainly characterized by the certificate of the server, the version of the protocol and the cipher-suite chosen to encrypt the data.

Website owners have been encouraged to switch from HTTP to HTTPS, and users have been taught to trust the contacted party upon recognizing HTTPS usage. As a matter of fact, a survey performed by Survata Consumer Research on behalf of the CA Security Council [2] revealed interesting findings. Out of 670 US consumers interrogated, 53% acknowledged that the presence of a padlock in the address bar was synonym of more trust, and 42% that the presence of a green bar meant greater safety. But despite that awareness concerning the security indicators, little is actually understood as to how the process is secured. The users' trust is often granted upon recognizing visible indicators, without a thorough check of the presented certificate.

Recent statistics highlight the slow shift from cybercriminals towards HTTPS [3], in order to exploit that gullibility. The shift is made even easier by the use of cloud services providing forms or web hosting services. The danger it represents is that the HTTPS features no longer guarantee the legitimacy of a particular website or content, especially for cloud services.As any content (legitimate or malicious) hosted on their domain

would borrow their certificate, this configuration is exploited by phishersin order to benefit from a valid (possibly an Extended Validation [4]) certificate for a reasonable fee (the hosting fee) without the hassle of going through certificate issuance procedures. The original purpose of our research was to determine to what extent parameters pertaining to the use of the HTTPS protocol are exploitable in the detection of fraudulent websites. Then, gaining advantage from these results, develop a system to protect users from phishing.

While we analysed both legitimate and phishing websites from the assumption that phishing websites are more likely to present suspicious SSL/TLS features, compared to legitimate ones, we concluded that those heuristics were not discriminant enough, due to the close profiles of phishing and legitimate sites. We then surveyed phishing websites abusing the users' excessive trust in SSL/TLS and we specifically considered the case of phishing websites hosted on hacked domains. To counter such websites, we designed a detection method based on their visual identity. Using this method, we were able to obtain an accuracy of 87% to 92% in the detection of this type of websites.

The rest of this paper is organized as follows: in Sect. II, we describe our evaluation of HTTPS phishing heuristics as well as the results obtained and the insight we gained from analysing HTTPS phishing websites. In Sect. III, we describe the method we implemented to detect HTTPS phishing websites hosted on hacked domains. Several related works on HTTP phishing detection methods are reviewed in Sect. IV and limitations associated with our detection method are discussed in Sect. V, before concluding in Sect. VI.

II. Motivation

As illustrated by the statistics introduced in the introduction, the presence of the SSL/TLS protocol represents a guarantee of security for a large amount of users. Various research works have considered the evaluation of the effective level of trust to grant to that technology. For example, Brubaker et al. [5] introduced a methodology to perform a large-scale evaluation of certificate validation logic in SSL/TLS libraries by leveraging random alterations made to real-world certificates.These crafted certificates allowed to evaluate rarely tested specifications, hence uncovering various security vulnerabilities in SSL/TLS and web browsers implementations. Levillain et al. [6] retrieved SSL/TLS data relative to HTTPS servers reachable between July 2010 and July 2011. An evaluation of the quality of SSL/TLS answers revealed some compliances issues with the standards, as well as the inaptitude of some servers to support some ciphersuites or latest protocols versions. In [7], Pukkawanna et al. assumed that a weak protocol version or algorithm is source of concerns, due to security flaws. A communication based on these parameters is thus deemed more insecure than one initiated with stronger protocol versions or encryption algorithms. Hence, an evaluation of the cipher-suites and protocol versions based on known vulnerabilities allowed to classify the servers as being secure, risky or insecure. That score, combined with the one obtained by an evaluation of the certificate provided the total score of the server, and allowed to classify approximately half of the servers as presenting security concerns.

The results of these studies are evidence that the notion of trusted servers should be considered with caution by the users, since it depends greatly on the care given to the implementation details of the platform and the scope of vulnerabilities discovered at the time. The works described previously considered the SSL/TLS ecosystem from a global standpoint. No distinction was made between servers hosting legitimate websites and servers hosting phishing ones. As a consequence, the behaviours of these two classes of sites could not be inferred from the results obtained. It could however prove to be a useful information for the detection of deceitful websites, since, as we discussed in the introduction, attackers are shifting towards HTTPS. We thus oriented ourselves towards the particular issue of HTTPS phishing with the aim to uncover discriminating SSL/TLS characteristics between phishing and legitimate pages and examined whether or not SSL/TLS features generate exploitable results in the detection of web-fraud.

A. Reminder: SSL/TLS operation

The following steps are performed during a SSL/TLS handshake:

- First, the security parameters to be used for the secure channel between the `Client` and the `Server` are negotiated via a `Client Hello` and a `Server Hello` message: the `Server` chooses among the supported parameters sent by the `Client` (cipher-suites, protocol versions, compression methods) those deemed acceptable
- Then, the `Server` sends its certificate to the `Client` for validation, using a `Server Certificate` message
- After this step, a `Client Key Exchange` is sent from the `Client` to the `Server`, containing the pre-master key generated by the `Client`. That key is used to generate the symmetric key allowing to encrypt the data
- The `CipherSpec Exchange` and `Finished` messages are finally sent by both parties to notify that the next messages will be encrypted.

From the point of view of the `Client`, the information obtained, and thus the features on which we base our assessment are the cipher-suites proposed by the `Server`, the versions of the SSL and TLS protocols in use, as well as the compression methods, and the `Server` certificate information.

B. A taxonomy of HTTPS phishing websites

In order to validate the discriminating effect of SSL/TLS handshake information to the detection of HTTPS phishing pages, we surveyed the usage of SSL/TLS features in several datasets of HTTPS webpages as described in Sect. **??**. To that end, our team began by building a taxonomy of HTTPS phishing webpages based on their hosting method as hinted by Pajares [3]. Two categories emerged: websites hosted on their own domains and websites taking advantage of third-parties (cloud services, hacked domains, shared SSL domains),

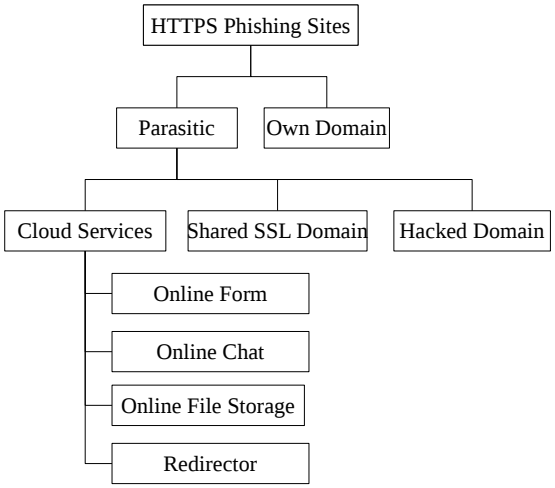

Fig. 1. Taxonomy of HTTPS phishing sites.

in order to give access to their contents.These two categories were further refined as illustrated in Fig. 1. Phishing websites hosted on their own domain have been identified using the following indicators:

- The reported phishing URL is a domain with no subfolder
- The domain name looks clearly incriminating (similarity with legitimate ones, presence of security-related keywords, well-crafted sub-domains)
- No other uses for the domain were found in search engines or archive.org

In that case, the SSL/TLS certificate presented is provided by the attacker, incurring further financial cost.

Parasitic sites are described as being hosted on a domain that does not belong to the phisher: a shared SSL domain, a cloud services platform or a hacked domain. That domain is most of the time shared with other legitimate users: all of them will present the certificate of the owner of the platform. Such method is usually less costly to the attackers, except for the one that requires pre-emptive compromise of an existing domain (*hacked domain*).

C. Analysis

Our original assumption was that malicious websites are less likely to present properly-filled certificates. But, for parasitic HTTPS phishing webpages, the information presented is not manipulated by malicious parties and is directly provided by the administrators of the hosting platform. Thus, it cannot be directly used to gain knowledge regarding the status of a webpage hosted on these services, even when qualified as *insecure* by the analysis module presented below.

In order to analyze the relation between the above-mentioned categories of HTTPS phishing webpages and their usage of SSL/TLS mechanisms, we collected several certificates and SSL/TLS information (protocol versions, cipher-suites) from known legitimate websites and known malicious websites. The provenance of the collected information helped us classify them among the different categories defined in our taxonomy. Finally, we leveraged the assessment of SSL/TLS servers previously performed by Pukkawanna et al. [7] as it is concerned with the security of such servers.

1) Datasets: We collected the following datasets:

- 1,213 legitimate certificates retrieved by connecting to Alexa Top 3,000 websites [8]: Legit_cert
- 1,170 phishing certificates retrieved by connecting to websites identified in Phishtank database [9]: Phish_cert
- SSL/TLS protocol and cipher-suite information of 103 online phishing websites: Phish_SSL
- SSL/TLS protocol and cipher-suite information of 102 online legitimate websites randomly selected among Alexa Top 3000 websites: Legit_SSL

The last two datasets have been obtained using the backward compatible Firefox configuration to contact the servers [10].

2) Classification: When connecting to a website, we extracted informations in the meta tag and searched for keywords indicating the use of a cloud service, such as : "online", "form", "shared", "storage", "cloud", ... On the other hand, we matched the Common Name (CN) in the retrieved certificate against a known list of shared certificates CNs, and identified the use of wildcard certificates. These steps allowed us to single out websites presenting shared certificates, the others being considered as presenting their own certificates.

3) Security assessment: Pukkawanna et al. [7] introduced two scores to assess a server according to the following criteria:

- the first group of criteria, relative to the information contained in the certificate, i.e., the presence of suspicious entries in the fields (see Table I).
- the second group, relative to the handshake information of an SSL/TLS server, specifically the protocol and cipher-suite chosen by the server. These parameters were classified as being secure, risky or insecure, according to known flaws and vulnerabilities.

According to these criteria, was identified as malicious a certificate with suspicious values (“SomeCity”, “SomeState”,...) or that lacked one of the following fields:

- Common Name, CN
- Organizational Unit, OU
- Organization, O
- Country, C
- State, S

However, a prior analysis of Legit_cert dataset showed that even secure certificates do not always include these 5 fields at once. 90% of the dataset is constituted of certificates presenting a subset of 2 or 3 of the previous fields. While Common Name is always provided, along with Organization or Organizational Unit, Country and State fields are most of

TABLE I
SUSPICIOUS VALUES IN THE CERTIFICATE. [7]

C (Country)	O (Organization)	OU (Organizational Unit)	S (State/province)	CN (Common Name)
XY, NON-STRING-VALUE, single/double quotation	SomeState, Someprovince, SomeOrganization, MyCompany, self-signed, 127.0.0.1, any compromised CA or cheap reseller CA	single/double quotation, Single dot, SomeState, Someprovince, SomeOrganizationUnit, Division, section, self-signed, 127.0.0.1, any compromised CA or cheap reseller CA	SomeState, Some-province, SomeState, Select one, Default, default	localhost.localdomain, 127.0.0.1

the time left blank. We adapted the script in order to include these use cases, identifying as insecure a certificate presenting suspicious values or less than two of the aforementioned fields. The results of this reviewed implementation are presented in Tab. II.

Besides, we only slightly modified the script provided in [7] in order to include the new cipher-suites we encountered during the collection of Legit_SSL and Phish_SSL. We relied on known flaws and vulnerabilities associated with a particular protocol version or cipher-suite to rate the servers. The results obtained after the analysis of Phish_cert and Legit_cert, and on Phish_SSL and Legit_SSL are presented in Tab. III and Table IV, respectively.

D. Results

From the results in Tab. III obtained via an analysis of Phish_cert, only 1.90% of phishing websites have been identified as being insecure after evaluating their certificates. Among these insecure certificates, 72.72% are own certificates. In order to be more accurate with the profile of legitimate websites we loosened the rules allowing us to label the certificates, however this fact is an impediment to our ability to detect insecure certificates too. Indeed, phishing and legitimate websites tend to share similar characteristics (presenting only the CN, OU and O fields) and hackers make an effort to present trustworthy values.

As shown in Tab. III and Tab. IV, more *insecure* servers can be encountered among the legitimate ones than among those that host phishing websites; however this last population presents more risky servers. Further investigations showed us that no insecure certificates have been presented by insecure or risky servers. Overall, only approximately 16% of servers (risky + insecure) hosting phishing websites have been identified as risky or insecure, the majority of them being servers presenting their own certificates. On the other hand, 9% of servers hosting legitimate websites have been identified as risky or insecure.

E. Discussion

In [7], servers from the whole IPv4 address space were considered in order to design a way to determine their level of security. However, the maliciousness of the service provided via SSL/TLS was not examined, as that information was not available in the original dataset.
In this work, we filled this gap by labelling the servers: we narrowed the scope to servers hosting legitimate and phishing websites reachable over HTTPS and computed for each of them the level of security of the certificate and of the protocol version and cipher-suite chosen. From this experimentation realised on the SSL/TLS handshake parameters presented by the servers, we conclude that heuristics specifically yielded by HTTPS cannot efficiently discriminate the legitimate websites from the phishing ones. The parameters examined show very little relevance in the detection, due to the close profiles of phishing and legitimate sites. As a consequence, the use of these heuristics will be more likely to generate a large number of false negative results.
On the other hand, by applying the taxonomy described earlier to our phishing dataset, we have also been able to identify the parasitic ones as being the most challenging and likely to evade current detection tools. They are on one hand, websites hosted on cloud services or forms generated by online platforms, and on the other hand, hacked websites used to host phishing web pages. Indeed, in the case of cloud services and online generated forms, the URLs are pre-formatted by the owners of the platforms and can appear suspicious, since they are longer than usual ones: they often contain long hexadecimal strings in order to identify the resource and its owner. Heuristics such as the history , popularity and creation date of the website are less likely to be relevant because they will yield seemingly trustworthy values. Besides, a reliable certificate is also going to be presented. Among our SSL heuristics, that last parameter is the one that is directly accessible and more likely to be the object of an assessment by a fraction of users, as we discussed in our introduction. However, by running our experiment, we have been able to realise that, the certificate might not only be evaluated as being safe (as per our analysis method), but it might also be source of confusion to identify the origin of a particular content presented. Indeed, by analysing the certificate, users might identify a content as originating from the legitimate owner of the platform and fall prey to fraud pages presented.
How is it possible to detect the malicious content in that case? In the following sections, we will focus on the particular issue

TABLE II
REPARTITION OF HTTPS PHISHING CERTIFICATES ACCORDING TO THEIR TYPES (TOTAL OF 1,170 CERTIFICATES).

Secure certificates %		**Insecure certificates %**	
98.1		1.9	
Own certificates %	**Shared certificates %**	**Own certificates %**	**Shared certificates %**
63.88	36.11	72.72	27.28

TABLE III
REPARTITION OF SERVERS HOSTING HTTPS PHISHING WEBSITES BASED ON THE CHOSEN PROTOCOL AND CIPHER-SUITE (TOTAL OF 103 SERVERS).

Secure servers %		**Risky servers %**		**Insecure servers %**	
83.5 %		12.62 %		3.88 %	
Own certificates %	**Shared certificates %**	**Own certificates %**	**Shared certificates %**	**Own certificates %**	**Shared certificates %**
51.16	48.84	76.92	23.08	100	0

TABLE IV
REPARTITION OF SERVERS HOSTING LEGITIMATE WEBSITES BASED ON THE CHOSEN PROTOCOL AND CIPHER-SUITE (TOTAL OF 102 SERVERS).

Secure servers %	**Risky servers %**	**Insecure servers %**
91.17	0	8.83

of hacked domains hosting phishing pages.

III. DETECTION OF HACKED DOMAINS HOSTING PHISHING PAGES

A. Assumptions

The main advice given to a user in order to identify a phishing page is to never trust the look and feel of the content he is presented, but rather rely on different indicators. However, what if that perception could actually be engineered in a different way to help the detection in the case of hacked domains?

Starting from the assumption that every website has its own visual identity allowing a user to distinguish it from another one, the fact of introducing a page impersonating a different website is more likely to disrupt that uniqueness. We were actually able to observe this characteristic among hacked domains present in the Phishtank database. If a user took the time to explore the website, instead of directly trusting the form presented on the page he landed, he could realize that incoherence. This way, by analysing the look and feel he receives not only from the first page presented, but from the global domain, he might be able to correctly identify a deceptive content.

Since it is not realistically possible to expect users to execute that analysis themselves, we aimed to introduce a system based on intelligent perception to automatically perform that behaviour. Thereby, being able to extract and compare the visual identity of the global website and the currently browsed URL gives a way to deem a website as being hacked, if too dissimilar results are obtained.

This approach distinguishes itself from the research performed by Zhang et al. [11], as well as Fu et al. [12] which both used visual analysis. We propose to compare a page to others hosted within the same website in order to get their level of similarity, and this way spot anomalies that could allow us to detect phishing pages mimicking an unknown target.

B. Perceptual Image Hashing

Hashing algorithms are used to convert files into a fixed-length string, representing the fingerprint associated with a particular input file. They refer in general to cryptographic hash algorithms, that allow to associate two files with a slight variation in the content, to hashes that are extremely distinct. However, in our approach, it is essential to carry the information relative to the similarity of the input files in the hash obtained. It is performed by the use of perceptual hashing, a different category of hashing algorithms applied on pictures. Perceptual hashing allows to maintain the correlation that exists between the inputted files by generating similar hashes for similar pictures. It is then possible to assess that level of similarity by the use of a comparison algorithm such as the Hamming distance on the generated hashes. In our system, we opted for the use of dHash [13], a perceptual hashing algorithm that realizes the following steps:

- reduction of the size of the picture
- conversion of the image to a gray scale picture
- computation of the difference between two adjacent pixels
- assignment of the bits based on whether the left pixel is brighter than the right one

Its output is then a hexadecimal string representing the hash of the image.

C. Proposed System - AJNA: Anti-phishing JS-based visual analysis

1) Overview: The system introduced in this paper considers the following scenario, described on Fig.2: The request sent

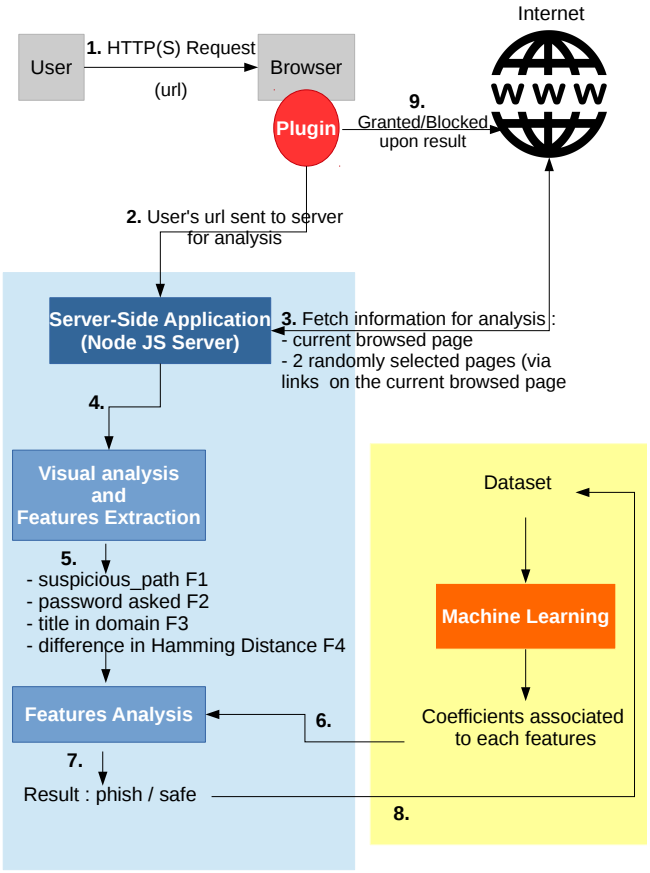

Fig. 2. Overview of the System.

by the user in order to access a website is intercepted by a plugin installed in the user's browser. That plugin retrieves the user's url and sends it to a Node JS server, on which runs our proposed application. That application issues a serie of requests in order to acquire the information needed for the analysis, that is the page currently browsed by the user, as well as two randomly selected pages on the same domain. These objects are then processed by the application in order to extract the features, and compute a result concerning the status of the page to be accessed, result that is sent back to the plugin, that decides wether or not to grant the navigation to the website, based on the received output. A machine learning phase is performed prior to the application analysis, in order to obtain the coefficients needed to compute the output. That output is also added to the machine learning dataset, allowing to enrich it over time.

2) Workflow: Server-side application: As presented, a Firefox plugin has been implemented. Its purpose is to send the currently browsed URL to an application running on a NodeJS server, responsible for running the visual analysis during the loading of the page.

Upon reception of the URL, the server-side application retrieves all the links present on the home page of that domain

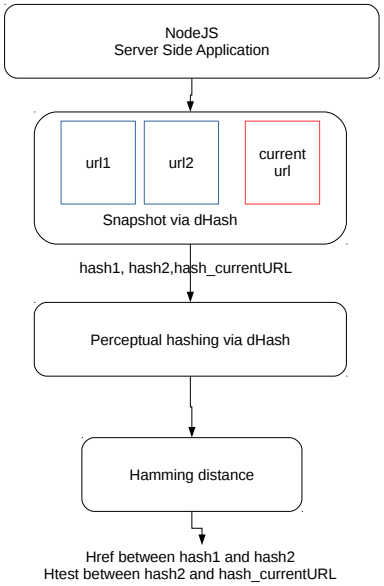

Fig. 4. Visual Analysis details.

and randomly selects two of them: URL1 and URL2.

These URLs, along with the currently browsed one are then passed to a headless browser (Phantom JS) that takes a snapshot of the three rendered pages. The window is calibrated to get only the top of the pages, where the menu bar is more likely to be localized; the text content of paragraphs and links is hidden before rendering.

For each of the snapshots, a hash is obtained by the use of dHash, the perceptual hashing algorithm we introduced in the previous section.

The computed hashes are compared using the Hamming distance in order to get their level of similarity:

- the Hamming distance between the hash of URL1 and the one of URL2 gives a reference on how dissimilar we can expect two pages on the same website to be : H_{ref}
- the Hamming distance between the hash of URL2 and the hash of the currently browsed URL gives us the divergence between a page of the website and the page we want to assess: H_{test}

The heuristic exploited is the difference between these two hamming distances: $(H_{test} - H_{ref})$. The graph on Fig. 3 shows the dispersion of this heuristic, according to the status of the website: phishing or legitimate. Fig. 4 illustrates an implementation of our mechanism.

In our configuration, we converted the image in a 16-bytes hash. An experimentation with lower and higher hash lengths (8, 24, 32) confirmed that value to be a right compromise, by containing enough information while limiting the dispersion.

Additional heuristics used: Along with the parameter

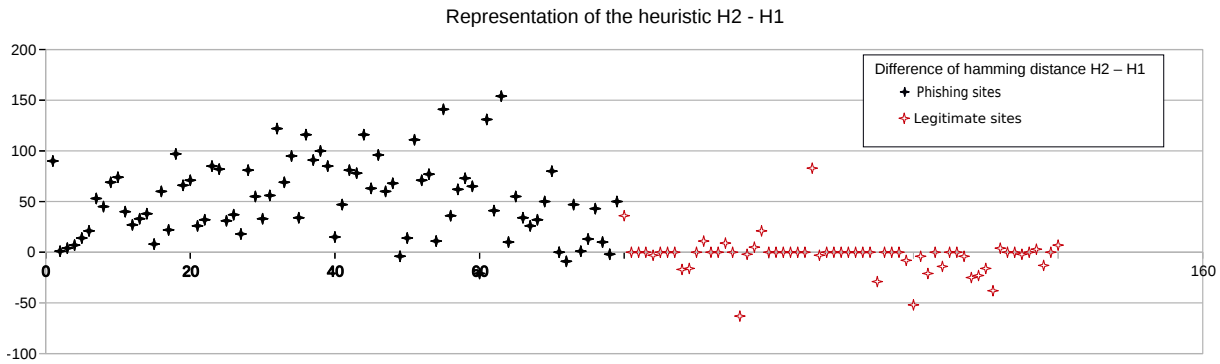

Fig. 3. Representation of the heuristic $(H_{test} - H_{ref})$.

TABLE V
CLASSES OF WORDS IN PHISHING URLS

Brands (b)	**Type of doc (tod)**	**Secure process (sp)**	**Display (dis)**	**Positive words (pos)**
google, usaa, paypal, drive, live, youtube, onedrive, tor, dropbox, bank, yahoo, apple, itunes, amazon, twitter	form, forms, survey, mail, share, file, online, images cloud, plugins, download, shared, files, documents, sharing, document	login, send ,secure, pay, update, sid, account, key, verification, sign, security, user , analysis, session, unsubscribe, accounts, confirm, client, sec support, submit, subscribe, protection, password, connect, confirmation, log, updates, billing, payment, verify, safe, upgrade, verifies	view, web, home, host, watch, content, index, web, hosting	important, welcome, insurance,promo, awesome, service, daily, tip, fresh, strong, good, review, hint, best, benefit, free,health, interesting, now

$(H_{test} - H_{ref})$, we introduced legacy features used in previous researches, but also new ones discovered during the analysis of our phishing dataset.

The first legacy features used is the **presence of forms asking for password or credit card information on the browsed page** (feature F1). We automatically identify a page without forms as being safe, and make sure to check not only the HTML source code, but the JavaScript source code as well.

The second legacy feature is **a suspicious URL** (feature F2). We based our categorization on the presence of an IP address, a suspicious port or character, the number of dots in the domain name, a suspicious redirect or path, as already used in past works. However, we also performed a survey of phishing URLs active between August 2014 and June 2015 that allowed us to obtain five classes of keywords recurrent in this population. The classes are reported on Table IV. On Fig. 5, we can have a better understanding of the distribution of these classes across the retrieved URLs. The different categories have been abbreviated as folows:

- Brand, b
- Type of doc, tod
- Secure process, sp
- Display, dis
- Positive words, pos

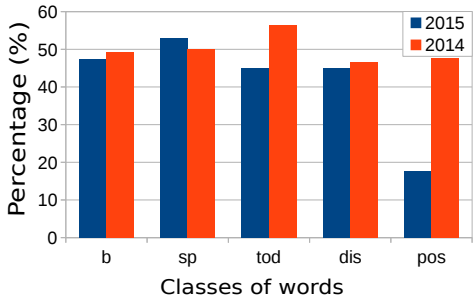

Fig. 5. Repartition of keywords in the phishing URLs.

We further examined the URLs in order to grasp how the five classes of words we identified are combined to each other, as illustrated on Fig. 6. We can notice that even if the proportions vary more or less over time, hackers will generally tend to use one or more of these categories in order to better deceive the users. Indeed, we can observe that only slightly under 14% of the URLs examined in both 2014 and 2015 do not have words contained in one of our classes. These classes we extracted will be one of the criterion to label a URL as being suspicious.
Besides, based on our observations, we also add the presence of a different domain in the path of the URL as a suspicion

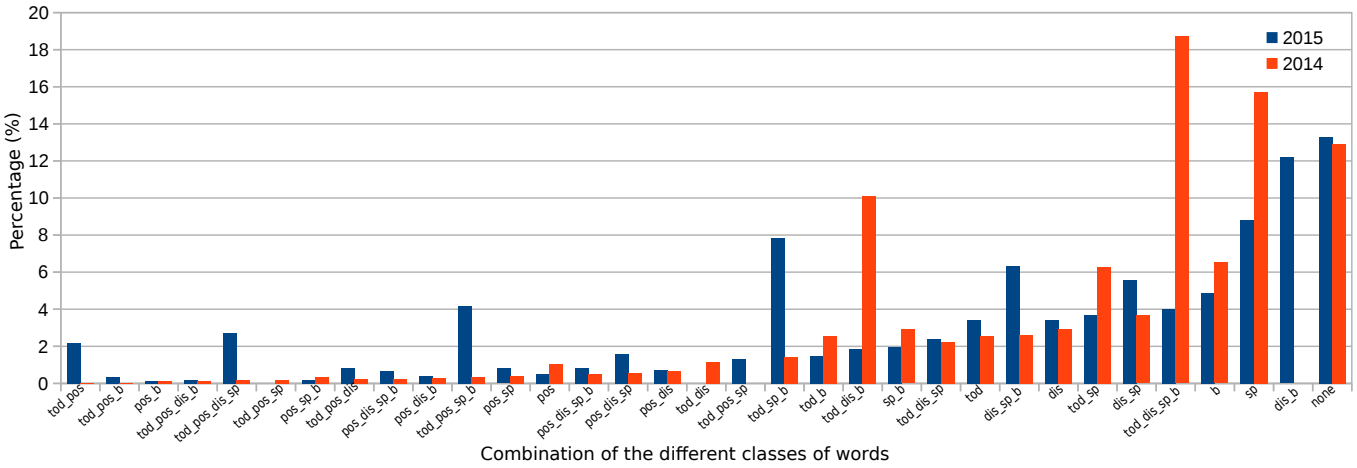

Fig. 6. Relationship between keywords in the phishing URLs.

criterion.

A novel feature derived from our analysis is **the presence of a portion of the title tag content in the domain name** (feature F3); that parameter is rather empirical. Indeed, we analysed the html source code of legitimate websites, and particularly the content of the title tag (<title></title>), also rendered on the top of the browser. We observed that legitimate websites tend to put their business name in the title tag, alongside various information such as the general purpose of the page or the title of an article presented, generally for Search Engine Optimization (SEO) purposes; besides, the domain name of the legitimate sites is also a reminder of the name of the brand. Consequently, it is possible to find a matching string between the title and the domain name of a legitimate website. On the other hand, phishing sites that often mimic this characteristic to better trick the users, and indicate the name of the targeted brand at this position(title tag), don't exhibit a domain name related to the impersonated brand. Hence, if a common string is not found between the domain name and the title, we are more likely to be on a phishing site.

These 3 features are subsequently fed to our classifier, along with our fourth feature F4 introduced earlier: $(H_{test} - H_{ref})$.

D. Machine Learning Phase

We collected the previous heuristics on a set of 140 websites:

- 70 websites gathered via Phishtank database, they represent hacked domains hosting phishing pages
- 70 websites being legitimate pages belonging to the Alexa Top 3000

A fluctuating acquisition time has been observed for each website: from 1 to 10 seconds for 70% of the sample, and up to 20 seconds for the rest. Further investigation allowed to pinpoint the rendering of the web page by the headless browser as being the bottleneck of the system. Indeed, the more extensive the text content is, the more processing time is needed, as it is necessary to discard that content to avoid the inclusion of more divergence in the snapshots. We explored ways to alleviate this shortcoming in Section V.

In order to understand the impact of each feature in the detection, we separated them into three sets:

- the first set, containing F1 and F2, the selected features used in previous works
- in the second set, we introduced our first heuristic F3 in addition to F1 and F2
- in the third set, we used the four heuristics F1, F2, F3 and F4

Facing a classification problem, we fed our dataset to the Support Vector Machine (SVM) algorithm, a simple machine learning classifier. SVM allows to find a decision boundary which separates the space in two regions. The hyperplan found is as far from all the samples as possible.
Our goal here was to train the classifier to discriminate between hacked domains and legitimate websites.
The 10-Fold cross validation method was used in order to evaluate the performance of the classifier: the dataset was randomly splitted into 10 sets, each one containing a training set of 126 websites and a testing set of 14 websites.
We ran this 10-Fold cross validation on each of the 3 sets of features previously introduced, and plotted the Receiver Operating Characteristic (ROC) curve for each of the experiments. The average of the metrics for each experiment can be observed on Table V. From the results in Table V, we can notice the evolution of the metrics, according to the chosen set. The more features we have in the set, better are the results.

TABLE VI
AVERAGE OF THE METRICS OBSERVED DURING 10-FOLD VALIDATION, ACROSS THE 3 SET OF FEATURES.

	TPR	TNR	FNR_score	FPR	Precision	Recall	Error	F1_score
Set 1	0.47	0.46	0.53	0.54	0.51	0.47	0.54	0.49
Set 2	0.54	0.58	0.46	0.42	0.71	0.54	0.44	0.62
Set 3	0.89	0.90	0.11	0.10	0.90	0.89	0.11	0.90

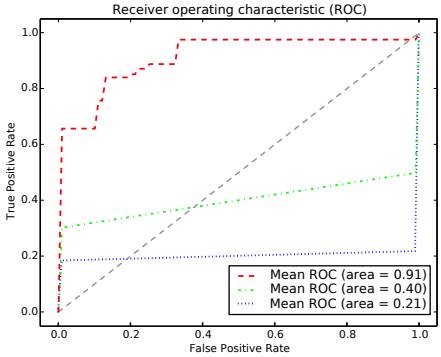

Fig. 7. Representation of the Mean ROC curve for the three sets of features.

F3 and F4, the parameters introduced in this study, allow to have a more accurate classification. Indeed, the best metrics are obtained for the third set, which combine F1, F2, F3 and F4, where the phishing detection accuracy reaches 90%. That is also observable on Fig. 7: we start from a mean Area Under the Curve (AUC) of 0.21 with the first set of features, to reach a mean AUC of 0.4 with the second set. The best perfomances are obtained with the third set of features where the mean AUC is equal to 0.91. We also used the Random Forest Regressor in order to rank each feature according to its impact in the detection. F4 ranked first, followed by F1, F3 and F2.

From this experiment, we can conclude that the new heuristics we introduced allow to discriminate between legitimate sites and hacked domains without having to depend on a database of known targets.

These results seem promising, even if they should be further confirmed with a larger set of examples. We believe the unconventional form of the ROC curves obtained for the set 1 and set 2 can be explained by a number of test samples too low. Indeed, the KFold, being performed on 10% of the dataset (14 samples), some experiment's rounds have a detection rate of 1, while others have a detection rate of 0: that configuration directly affect the average recall and precision represented on Fig.7.

IV. RELATED WORK

A. Phishing detection methods

The techniques used to combat phishing can be organized in two classes: non technical and technical methods. While the former have the user as principal actor of the detection process and seek to train him, the latter perform an automatic analysis without requiring the user's intervention.

1) Non technical methods: They are generally designed with the education of the user in mind; since he is ultimately responsible for revealing or not his credentials on a malicious website.

Non technical methods can take the form of games or educative materials where extensive details are given about how to recognize phishing websites and adopt safe behaviors. An example of educational material is the Phishing Education Landing Page from the Anti Phishing Working Group (APWG) [14]. This initiative is a collaboration between the APWG and corporations that have been targeted by a phishing campaign. The known malicious URL is redirected to a safe page where the user is presented with clues allowing him to recognize malicious websites.

Arachchilage et al. [15] and Hale et al. [16] oriented themselves towards the game approach. They place the user in a fictitious environment where he faces use cases designed to be close to real life ones and has to identify whether a web page is legitimate or not, based on various features; in that process, the aim is to instil the habit to check these parameters before trusting any website.

Non-technical methods generally have the advantage to impact instantly the user by training him and giving him clues on how to assess the legitimacy of a website. Subsequent evaluations show an improvement of the detection rate of phishing websites by the users right after the training.

However, on the downside, the teachings tend to fade away if not regularly repeated.

2) Technical methods: They often take the appearance of browser-side anti-phishing methods.

Blacklists approaches leverage services or databases that register known-phishing URLs. For example, browsers such

as Firefox and Google Chrome offer Google Safe Browsing as a built-in tool [17]. This technology uses blacklists compiled by Google while crawling the web, in order to warn the users when they access phishing websites. The Opera browser on the other hand uses a blacklist compiled with the ones of Phishtank, Netcraft, as well as TRUSTe [18]. In addition to blacklists databases, Netcraft Toolbar, a browser plugin, also provides protection against not yet registered phishing websites by computing a risk rating. That score is based upon the history of phishing sites in the same domain, a suspicious URL or port number, the hosting history of the ISP, the country, or the top-level domain history, regarding phishing websites. They also consider the site popularity among other Netcraft users [19]. On the other hand, whitelists approaches can also be combined to blacklists ones. By referencing a subset of trusted websites that will evade filtering, they allow to reduce the potential number of false positives.

Previous works have been done by Zhang et al. [11], but also by Fu et al. [12], exploiting the visual aspect of a website. Both used the visual similarity, computed via perceptual hashing or the earth movers distance algorithm, to make the distinction between phishing and legitimate websites. They used a set of known phishing templates as reference for the comparison. However, these methods could not detect new impersonated brands as they relied on the use of a database of known targets. Still using visual analysis, Corbetta et al. [20] interested themselves to the detection of fraudulent websites by the identification of counterfeited or misappropriated certification seals displayed on a page. Comparing images displaying potential seals to known seals via the use of perceptual hashing, they have been able to spot misused stamps, and hence deceitful sites.

The advantage of the blacklist approach is that chances are very high that the website registered is indeed a phishing one. However the main issue is its inability to recognize not yet blacklisted sites, thus leaving the user vulnerable to them. Besides, whitelists also represents a double-edge sword. Indeed, a domain compromised or hosting a malicious content (in the case of cloud platforms) after its addition to the list, will escape the detection and still remain trustworthy to the users.

On the other hand, heuristic-based methods present a more dynamic approach as they are able to spot phishing websites that are not yet registered in a blacklist. Several heuristics have been developed based on the URL, the content, the ranking or the DNS features in order to make the distinction between phishing websites and legitimate ones. Ultimately these heuristics can be combined with machine learning algorithms in order to address the polymorphism and versatility of phishing websites.

In [21], Aburrous and Khelifi classified the features into 6 groups: URL and domain identity, security and encryption, JavaScript source code, page style and contents web address bar, and social human factor. These features are then used to extract classification rules by leveraging fuzzy logic and data mining associative classification.

Huh and Kim [22] used the reputation of a website as heuristic. That heuristic is obtained via the use of popular search engines. A comparison between machine learning algorithms is subsequently made in order to determine the one with the best performances.

V. Discussion

In this work, we have presented the dangers inherent to an excessive trust into the SSL/TLS indicators, i.e., unsuspecting users blindly trusting a webpage when it advertises its use of an SSL/TLS certificate. Security indicators, such as the padlock icon, have been criticized in the past to be difficult to interpret by users as they often lack the knowledge of security indicators [23], and are now more confusing to them since they are also misused by attackers to convey a false sense of security [24]. Mobile users are even more oblivious of their presence when dealing with the reduced screens of their smartphones [25]. Indeed, there is no objection to the increased security brought by using SSL/TLS in HTTP communications, and research works specifically aiming to educate users into recognizing security icons [26] should be encouraged. But even security-savvy users may fall victims to phishing traps if they do not use their discernment when dealing with seemingly secure websites. It was already demonstrated by Pukkawanna et al. [7] that even websites presenting an SSL/TLS certificate could not guarantee the security of their transactions with the end-user.

We then distanced ourselves from these heuristics to propose a detection method able to deal with more evasive phishing pages in that they thrive upon legitimate domains. While we have demonstrated promising results, we have detected limitations with the actual implementation, i.e., the processing time incurred by the visual analysis has turned out to be the major bottleneck. Indeed, for every URL, three pages are fetched and processed to get the status of the website. This first implementation was more of a proof of concept and not oriented towards optimization. But we believe that this delay can be somehow reduced: taking advantage of the asynchronous capability of NodeJS, processing can be parallelized and distributed across a pool of headless browsers. The browsed URL could also first be compared with a database of known targets and if no match is found, the program could step into the processing of the snapshots to be sure that it is not facing an unknown target. On another hand, speeding up the processing time might allow to consider more than one Hamming distance as ground truth and find a better balance to get the visual identity of the website.

Besides, some websites present ads that vary according to the browsed page, or menus whose color adapts according to the addressed topic. These facts cause the generation of dissimilar hashes and thus a suspicious Hamming distance for two pages coming from the same origin. By introducing a list of resources that should not be fetched by the headless

browser, it could be possible to prevent the rendering of ads. As for color-themed menus, i.e., webpage menu bars that changed themes according to the page topic, instead of picking two URLs randomly on the landing page, it might be more relevant to classify the URLs in order to select those that appear to be referring to the same topic. This categorization can be achieved by grouping the URLs according to the first element of the path. Indeed, news sites like the one of the BBC, tend to organize articles pertaining to the same topic on different folders: news, sport, culture..., folders whose name will appear as the first element of the path.

Additionally, the visual perception method we have used is only effective in the case of hacked domains, where phishers would host a phishing page of a given brand on a domain they happened to have compromised, of which brand is different from the impersonated one. As no SSL/TLS heuristics are used, this method can be extended to HTTP phishing websites. It can be used in a more global phishing detection/mitigation system where it would specifically tackle phishing pages hosted on hacked domains. As mentioned in Section II-B, we have also identified other types of phishing pages that proved to be impervious to detection, such as pages hosted on cloud services or forms services platforms. A more straightforward approach would analyse the contents form the page in order to identify keywords related to form fields or submission, as well as sensitive information. Indeed, our work limited itself to the parameters presented by the phishing webpage and its URL, but more contextual information could actually be gathered from extending the detection workflow to blacklists and/or email clients from which most phishing emails originate.

The machine learning could benefit from a larger set of samples in order to validate the results. On the other hand, concerning the feature F1 (suspicious path), it would be more relevant to introduce a variable parameter, instead of a binary one. Indeed, in the current implementation, as soon as a suspicious rule is fulfilled, we automatically consider the path as suspicious. A better approach would be to rank the rules and score accordingly.

VI. Conclusion

In the survey presented in Section II, we evaluated the relevance of some of the heuristics proposed by Pukkawanna et al. [7] in detecting HTTPS phishing and came to the conclusion that solely based on these heuristics, we would not be able to discriminate phishing webpages from legitimate ones, ruling their direct integration into a detection system. Suspicious patterns in fields such as `DN`, `OU` and `O` would result unsuccessful at singling out phishing websites. Our additional survey over keywords has proven complementary to previous survey works on URL-based impersonation [27] where strings as manipulated to *look* like the target ones by means of suppressing a letter, using numbers in place of letters, etc. Combining the approaches would improve detection based on certificate information. By studying certificate information, cipher-suites and protocol versions chosen by both legitimate and phishing websites, we came to the conclusion that those heuristics were difficult to use to detect malicious pages.
However, among the group of HTTPS websites challenging current detection methods (parasitic ones), we further analyzed hacked websites and designed a detection method. A comparison of the visual identity of the website with the one of the browsed URL, if too dissimilar, allowed to identify compromised domains with a precision reaching 90%, without relying on an existing database.

Acknowledgment

This research has been supported by the Strategic International Collaborative R&D Promotion Project of the Ministry of Internal Affairs and Communication, Japan, and by the European Union Seventh Framework Programme (FP7/2007-2013) under grant agreement No. 608533 (NECOMA). The opinions expressed in this paper are those of the authors and do not necessarily reflect the views of the Ministry of Internal Affairs and Communications, Japan, or of the European Commission.
The authors would like to thank Pawit Pornkitprasan as well as Sirikarn Pukkawanna for providing their research on the taxonomy of HTTPS phishing websites. Our gratitude also goes to Olivier Levillain for his insightful comments.

References

[1] FBI, "IC3 Annual Report," https://www.fbi.gov/news/news_blog/2014-ic3-annual-report, 2014, [consulted 2015/09/13, Online].
[2] CA Security Council, "2015 consumer trust survey - casc survey report," https://casecurity.org/wp-content/uploads/2015/04/CASC-Consumer-Survey-Report-2015.pdf, [consulted 2015/09/13, Online].
[3] P. Pajares, "Phishing Safety: Is HTTPS Enough?" http://blog.trendmicro.com/trendlabs-security-intelligence/phishing-safety-is-https-enough/, [consulted 2015/09/13, Online].
[4] Global Sign, "What is an Extended Validation Certificate?" https://www.globalsign.com/en/ssl-information-center/what-is-an-extended-validation-certificate/, [consulted on 2015/10/30, Online].
[5] C. Brubaker, S. Jana, B. Ray, S. Khurshid, and V. Shmatikov, "Using Frankencerts for Automated Adversarial Testing of Certificate Validation in SSL/TLS Implementations," 2014.
[6] O. Levillain, A. Ebalard, B. Morin, and H. Debar, "One Year of SSL Internet Measurement," in *Proceedings of the 28th Annual Computer Security Applications Conference (ACSAC 2012)*, 2012.
[7] S. Pukkawanna, Y. Kadobayashi, G. Blanc, J. Garcia-Alfaro, and H. Debar, "Classification of SSL Servers based on their SSL Handshake for Automated Security Assessment," in *Proceedings of the 3rd International Workshop on Building Analysis Datasets and Gathering Experience Returns for Security (BADGERS)*, 2014.
[8] http://aws.amazon.com/alexatopsites/, [consulted 2015/09/13, Online].
[9] http://phishtank.com/, [consulted 2015/09/13, Online].
[10] https://wiki.mozilla.org/Security/Server_Side_TLS, [consulted 2015/09/13, Online].
[11] W. Zhang, G. Zhou, and X. Tian, "Detecting Phishing Web Pages Based on Image Perceptual Hashing Technology," *International Journal of Advancements in Computing Technology 4.2*, 2012.
[12] A. Y. Fu, W. Liu, and X. Deng, "Detecting phishing web pages with visual similarity assessment based on earth mover's distance (EMD)," *Dependable and Secure Computing, IEEE Transactions on 3.4*, pp. 301–311, 2006.
[13] N. Krawetz, "Kind of like that," http://www.hackerfactor.com/blog/?/archives/529-Kind-of-Like-That.html, [consulted 2015/09/13, Online].
[14] Anti Phishing Working Group, http://phish-education.apwg.org/r/en/index.htm, [consulted 2015/09/13, Online].

[15] N. A. G. Arachchilage, S. Love, and M. Scott, "Designing a Mobile Game to Teach Conceptual Knowledge of Avoiding Phishing Attacks," *International Journal for e-Learning Security 2.2*, pp. 127–132, 2012.
[16] M. L. Hale, R. F. Gamble, and G. Philip, "CyberPhishing: A Game-based Platform for Phishing Awareness Testing," in *System Sciences (HICSS), 2015 48th Hawaii International Conference on. IEEE*, 2015.
[17] Google, "Safe Browsing API," https://developers.google.com/safe-browsing/developers_guide_v3, [consulted 2015/09/13, Online].
[18] Opera, "Opera Fraud Protection," http://www.opera.com/help/tutorials/security/fraud/950/, [consulted 2015/09/13, Online].
[19] Netcraft, http://toolbar.netcraft.com/help/faq/, [consulted 2015/09/13, Online].
[20] J. Corbetta, L. Invernizzi, C. Kruegel, and G. Vigna, "Eyes of a Human, Eyes of a Program: Leveraging Different Views of the Web for Analysis and Detection," in *17th International Symposium, RAID 2014, Gothenburg, Sweden*, 2014.
[21] M. Aburrous and A. Khelifi, "Phishing Detection Plug-In Toolbar Using Intelligent Fuzzy-Classification Mining Techniques," in *The International Conference on Soft Computing and Software Engineering (SCSE'13)*, San Francisco State University, San Francisco, California, USA, 2013.
[22] J. H. Huh and H. Kim, "Phishing detection with popular search engines: Simple and effective," in *Foundations and Practice of Security*, S. B. Heidelberg, Ed., 2012, pp. 194–207.
[23] R. Dhamija, J. Tygar, and M. Hearst, "Why Phishing Works," in *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. ACM, 2006, pp. 581–590.
[24] A. Adelsbach, S. Gajek, and J. Schwenk, "Visual Spoofing of SSL Protected Web Sites and Effective Countermeasures," in *ISPEC*, vol. 3439. Springer, 2005, pp. 204–216.
[25] C. Amrutkar, P. Traynor, and P. C. van Oorschot, "Measuring SSL Indicators on Mobile Browsers: Extended Life, or End of the Road?" in *Proceedings of the 15th International Conference on Information Security*. Springer-Verlag, 2012, pp. 86–103.
[26] D. Miyamoto, T. Iimura, G. Blanc, H. Tazaki, and Y. Kadobayashi, "EyeBit: Eye-Tracking Approach for Enforcing Phishing Prevention Habits," in *Proceedings of the 3rd International Workshop on Building Analysis Datasets and Gathering Experience Returns for Security (BADGERS)*, 2014.
[27] P. Agten, W. Joosen, F. Piessens, and N. Nikiforakis, "Seven months' worth of mistakes: A longitudinal study of typosquatting abuse," in *Proceedings of the 22nd Network and Distributed System Security Symposium (NDSS 2015)*, 2015.