BULGARIAN ACADEMY OF SCIENCES

CYBERNETICS AND INFORMATION TECHNOLOGIES • Volume **18**, No 4
Sofia • 2018 Print ISSN: 1311-9702; Online ISSN: 1314-4081
DOI: 10.2478/cait-2018-0047

How Phishing Pages Look Like?

A. Bartoli, A. De Lorenzo, E. Medvet, F. Tarlao

Dipartimento di Ingegneria e Architettura, University of Trieste, Italy
E-mails: bartoli.alberto@units.it mimmuz2k5@gmail.com emedvet@units.it ftarlao@gmail.com

Abstract: *Recent phishing campaigns are increasingly targeted to specific, small population of users and last for increasingly shorter life spans. There is thus an urgent need for developing defense mechanisms that do not rely on any forms of blacklisting or reputation: there is simply no time for detecting novel phishing campaigns and notify all interested organizations quickly enough. Such mechanisms should be close to browsers and based solely on the visual appearance of the rendered page. One of the major impediments to research in this area is the lack of systematic knowledge about how phishing pages actually look like. In this work we describe the technical challenges in collecting a large and diverse collection of screenshots of phishing pages and propose practical solutions. We also analyze systematically the visual similarity between phishing pages and pages of targeted organizations, from the point of view of a similarity metric that has been proposed as a foundation for visual phishing detection and from the point of view of a human operator.*

Keywords: *Security, phishing, human factors.*

1. Introduction

Phishing is still a central security issue on the Internet. The most common and widely used defense mechanisms are based on forms of *reputation*: when a user attempts to visit a web site or IP address that is known to host phishing campaigns, access is automatically inhibited or made possible with an explicit warning to the user. Such defense mechanisms are integrated in modern browsers and are often deployed in web application firewalls that monitor outbound connections at the border of organizations. The reputation of web pages and IP addresses is based on information shared automatically by several actors on the Internet which systematically collect and share information about ongoing attacks. This way, when an actor detects an attack campaign based on pages at a certain URL or IP address, the whole community learns of that specific attack and may inhibit access to the corresponding location quickly. Furthermore, sharing and correlating information about suspicious activities improve the ability of detecting attacks for the whole community.

Recent attack campaigns tend to be more targeted to specific and small population of users and last for increasingly shorter life spans [1, 2]. Indeed, the most sophisticated campaigns often rely on phishing pages crafted for a single target (*spear phishing*). There is thus an urgent need for developing additional phishing defense mechanisms that do not rely on any forms of blacklisting or reputation: there is simply no time for detecting novel phishing campaigns and notify all interested organizations quickly enough.

A very promising framework in this respect consists in detecting phishing pages based solely on their *visual features*. The framework is based on an image classifier embedded in the browser and equipped with prior knowledge of the legitimate ⟨protocol, domain name⟩ pair(*s*) of each website of interest to the user. When the browser has loaded a web page *p*, the classifier determines whether the screenshot of *p* belongs to one of the visual classes corresponding to each website to be protected. In case of a match, the tool compares the actual ⟨protocol, domain name⟩ of *p* to those expected for that website and warns the user in case of a mismatch. The resulting defensive mechanism would implement the procedure that any technically-savvy and constantly vigilant user applies in practice, except that in this case the procedure would be automated and thus available to every user and continuously. The resulting scenario would thus raise the bar for attackers considerably.

There have been several proposals in the literature for actually implementing a framework of this kind [3-6], based on features extracted from web pages with image processing techniques (e.g., [7-9]). Recent advances in image classification based on deep learning could open new directions of research and enable practical solutions [10]. One of the major impediments to research in this specific, important area is the lack of systematic knowledge about how phishing pages actually look like. Are they an exact replica of the targeted page? Is a phishing page available at every resolution supported by a browser? Is a phishing page able to display correctly both in a smartphone and in a desktop? While the literature abounds of reports providing quantitative indexes about observed campaigns, we are not aware of any systematic analysis of the visual properties of phishing pages. In particular, we are not aware of any publicly available dataset of screenshots, which is an essential resource for any research and development effort is this area.

In this work we investigate this important issue. Our contribution is as follows: first, we describe in full detail the procedure for constructing a *large dataset* of *screenshots* of *phishing* pages. With this procedure we have collected and labelled more than 800 pages and 23,000 screenshots at different resolutions. We cannot release this dataset publicly because, unfortunately, doing so would violate the terms of service of the targeted companies. However, we believe that describing the technical problems for collecting a dataset of this kind, along with the corresponding solutions, is a valuable contribution. Second, we *quantify* the *visual similarity* between phishing pages and the corresponding targeted pages with a similarity metric that has been proposed as a powerful foundation for visual phishing detectors [5, 11]. This similarity metric, called Normalized Compression Distance (NCD) [12] is based on information theory and has been applied successfully in a number of different application domains [13-15]. Third, we compare the visual similarity quantified by

NCD to visual similarity as perceived by a *human operator*. According to our analysis, pages that are very similar in terms of NCD are indeed very similar visually. Unfortunately, the opposite is not true: pages that are not similar in NCD may look very similar to a human operator. This fact may constitute a serious weakness of using NCD as a foundation for visual phishing detection, because a detector of this kind would not be able to warn the user of a phishing attempt. We complete our study by showing a simple way for crafting a (phishing) page that looks very similar visually to the targeted page but very different in terms of NCD, thereby circumventing NCD-based visual phishing detectors.

2. Dataset acquisition

2.1. Data source and technical problems

We collected our dataset from Phishtank (**https://www.phishtank.com**). Phishtank is a service in which users submit suspected phishing pages and other users verify whether the submitted page is a real phishing attempt. The information made available by Phishtank is used by a number of defensive products: as soon as a phishing page is verified, all such products block access to that page. Phishtank Provides a Downloadable Dataset (PDB) of *phishing page entries*, updated once per hour. Each PDB entry contains: phishing page URL; name of the company that the phishing page attempts to impersonate; two flags that indicate if the phishing page is *verified* and still *online*. A page URL submitted to Phishtank becomes verified after other Phishtank users have visited that URL and decided that the page is indeed a phishing page.

Company name “Other” is used by default for entries in which a more specific company name has not been assigned. Company names are not assigned with perfect recall, i.e., an entry could be labelled “Other” even though that entry corresponds to a page attempting to impersonate company name “Facebook”. Company names are not assigned with perfect precision either, i.e., entries may be occasionally labelled with wrong company names.

Each PDB version contains about 30K entries and Phishtank contributors insert about 200 new entries per day. Collecting a dataset with many screenshots of several companies is significantly difficult and hard to automate, though:

- Most entries (about 75%) are associated with company name “Other”.
- Most of the entries (about 80%) that are associated with a company name target either Facebook or Paypal.
- Most of the screenshots (about 70%) that may be acquired by downloading PDB and then connecting to verified URLs are not usable for a screenshot dataset because they represent pages that are either:
    - blocked (by the browser, or by some defensive mechanism along the path between the browser and the web server hosting the phishing page), or
    - disabled (by the web hosting provider), or
    - labelled with a wrong company name, or
    - malfunctioning, or

○ useful as a phishing sample only at some browser resolutions (e.g., when rendered for a desktop but not for a smartphone or vice versa).

In other words, few new screenshots can be collected each day and the distribution of such screenshots over companies tends to be very sparse, except for very few heavily targeted companies.

Blocked and disabled pages are particularly relevant and unavoidable because many defensive mechanisms are based on monitoring the content of PDB itself and reacting as soon as a new entry is inserted. The figures above are estimates that we obtained in several preliminary acquisition campaigns. Their relevance is not in the exact value of each figure but rather in the underlying, intrinsic problem.

Based on these problems, we carefully designed a dataset acquisition procedure aimed at collecting tens of URLs for several companies, with 30 different resolutions for each URL. We describe such procedure and its result in the next section.

2.2. Dataset acquisition procedure

We acquired the phishing page screenshots by means of a procedure repeated each hour, as follows.

Step 1. Download PDB; let PDB_0 denote the last version of PDB that we downloaded.

Step 2. Remove from PDB_0 entries that were not flagged as verified and online, entries already processed in earlier iterations and entries whose URLs are duplicate of URLs in other entries (PDB occasionally contains such entries).

Step 3. Construct a temporary list T-PDB containing all remaining entries of PDB_0, ordered in a per-company stratified way; for companies with more than 1000 already acquired URLs, insert into T-PDB only 5 entries.

Step 4. Process each entry *e* in T-PDB as follows (*e.u* denotes the corresponding URL):

a. If *e.u* is hosted at an IP address owned by CloudFlare, then discard *e* and skip to the next entry (the reason for this step is explained below);

b. Download *e.u*; in case of any download error (e.g., TCP connection error, TLS error, HTTP 400,500 codes), then discard *e* and skip to the next entry;

c. Acquire 30 screenshots of *e.u* in a headless-mode Google Chrome browser; 15 different screen resolutions in desktop-mode, and 15 different screen resolutions in mobile-mode.

In case of any errors during the acquisition of a snapshot, retry the acquisition up to three times and skip the acquisition of the snapshot if the error persists; in case 10 snapshots have been skipped, skip to the next entry. Errors in a snapshot acquisition are detected either as an exception raised by the headless browser or as a fully white rendered page.

Step 5. Terminate the procedure after one hour and restart from Step 1.

We iterated the described operations between 23 July 2018 and 4 August 2018. While downloading the dataset, we also attempted to estimate the *life time* of a phishing page, i.e., for how long the page is made available by the server on which the page is hosted. For completeness of analysis, we provide our estimation methodology and results in the Appendix.

We disabled the integrated warnings for dangerous and deceptive web pages in the headless Google Chrome browser, otherwise the browser would have not rendered the actual content of the phishing page. We executed the above procedure (i.e., all the DNS and HTTP traffic) through the TOR anonymity network, in order to bypass the protection mechanisms on the frontier of our University network that would have prevented the rendering of several phishing pages.

Visual inspection of a sample of the acquired screenshots indicated the presence of a number of pages in which some of the components could not be downloaded and pages that were correctly downloaded but that displayed a form of "content removed" or "user banned" message. For this reason, we checked *all* the acquired screenshots visually in order to:

- Discard screenshots without any element for user interaction (form, buttons, links).
- Discard screenshots displaying error messages.

During this step we also:

- Discarded screenshots from entries annotated in Phishtank with wrong company names.
- Imposed an upper bound to the number of URLs for each company to 60 (i.e., once this quantity was reached for a given company, we skipped the visual inspection of any further screenshot of that company).

The reason for the check on whether the page is accessed through CloudFlare (Step 4a) is the following. CloudFlare is a company that offers services for improving performance and security of web sites (**https://www.cloudflare.com**). Such services are implemented by means of proxy servers placed in between users and web sites. Such a placement actually occurs at the DNS level, that is, by letting the name of the web site map to the IP address of a CloudFlare web server. End users will not notice that they are accessing the web site through CloudFlare, except that they will experience better performance and security. During preliminary experiments, we found that usage of CloudFlare is common in Phishtank-indexed pages, probably because those pages are located on large web hosting infrastructures that use CloudFlare services. Most importantly, we found that CloudFlare is quite fast in replacing Phishtank-indexed pages with either interstitial pages or pages displaying error messages (probably because CloudFlare blocks pages indexed by Phishtank). By discarding URLs that have to be accessed through CloudFlare, thus, we greatly decreased the number of pages to be checked visually without significantly decreasing the number of screenshots useful for our dataset.

The following table summarizes the resulting composition of our dataset, along with a summary of the screenshots that either have not been acquired or have been discarded after a visual inspection ("Missing screenshots" column) and of the URLs for which there is not even a single screenshot available ("Missing URLs" column).

It can be seen that for 31 URLs (3.8% of the total) we could not acquire any screenshot. The number of screenshots that could not be acquired was 526, corresponding to 2.27% of all the acquired screenshots. It can be seen that all the missing screenshots in desktop mode are associated with only three targeted companies, which suggest that such mistakes were present only in a few specific phishing campaigns. Thus, these data tend to confirm the intuition that phishing pages tend to visualize correctly on desktops. On the other hand, the missing screenshots in mobile mode are much more numerous than in desktop mode (481 vs 45) and more spread across targeted companies. Thus, these data suggest that the crafting of phishing pages for mobile screens tends to be less accurate than for desktops.

Table 1. Composition of collected dataset

Company	Number of URLs	Number of Screenshots	Missing URLs		Missing screenshots	
			Desktop	Mobile	Desktop	Mobile
ABSA Bank	60	1788	0	0	0	12
Dropbox	60	1751	2	0	0	19
Orange	60	1764	1	0	0	21
PayPal	60	1668	5	0	0	57
Yahoo	60	1777	0	0	0	23
Microsoft	59	1707	0	0	9	54
Facebook	56	1624	0	1	13	28
Google	53	1580	0	0	0	10
Adobe	51	1469	3	0	0	16
MyEtherWallet	39	1145	1	0	0	10
Alibaba.com	36	1060	0	0	0	20
eBay, Inc.	36	1080	0	0	0	0
AOL	27	734	2	0	23	23
Binance	20	443	7	0	0	52
DHL	18	508	1	0	0	17
Itau	14	346	3	0	0	29
JPMorgan Chase and Co.	14	385	2	0	0	5
ASB Bank Limited	13	388	0	0	0	2
Internal Revenue Service	13	390	0	0	0	0
Apple	11	309	0	0	0	21
Amazon.com	9	227	1	0	0	28
Netflix	9	213	2	0	0	27
Banco De Brasil	7	208	0	0	0	2
U.S. Automobile Association	7	209	0	0	0	1
Bank of America Corporation	5	150	0	0	0	0
Bradesco	4	120	0	0	0	0
Wells Fargo	3	86	0	0	0	4
Santander UK	1	30	0	0	0	0
Total	**805**	**23159**	**30**	**1**	**45**	**481**

3. Normalized compression distance

3.1. Motivation and definition

In order to gain systematic insights into phishing pages, we assessed the *visual similarity* between phishing pages and original pages. To this end, we followed the approach proposed in [5, 11], which quantified the visual similarity between a pair of web pages by means of the NCD [12] between the rendered images of those pages (full details of this computation are provided below). The experiments in [5, 11] demonstrated that such a quantification of visual similarity is both effective and useful in the context of visual detection of phishing pages.

NCD is a similarity metric that has been used successfully in a number of broadly different application domains [13-15] and is defined in terms of a compression algorithm C. Given two byte strings x, y, let xy denote their concatenation; let $C(z)$ denote the length of string z compressed by C. The NCD between x and y, denoted $\mathrm{NCD}(x, y)$ is defined as:

$$\mathrm{NCD}(x, y) = (C(xy) - \min(C(x), C(y))) / \max(C(x), C(y)).$$

The intuition is that if x and y are very similar to each other, then compressing their concatenation will produce a byte string whose length is almost identical to the length of either of the two strings.

The rationale of the approach in the context of phishing detection is the following. A visual phishing detector embedded in a browser is equipped with a set V of pairs $\langle v, \mathrm{URL}(v)\rangle$, where v is a rendered web page and $\mathrm{URL}(v)$ its identifier. Whenever the browser has rendered a page u:

1. The detector computes the NCD between u and each page in V;
2. If this value is below a predefined threshold, then the detector compares $\mathrm{URL}(u)$ and $\mathrm{URL}(v)$;
3. If these identifiers are different, then u is a phishing attempt targeting v and a form of warning is exposed to the user.

Although this basic idea may be augmented in several ways (e.g., the details of the comparison between $\mathrm{URL}(u)$ and $\mathrm{URL}(v)$, usage of a dynamic threshold or of a classifier based on NCD), the basic framework is appealing because it automates many of the actions that a technically-savvy user usually applies for defending him/herself from phishing attacks. We refer the reader to the cited papers for full details (including the potential latency and scalability issues intrinsic in the approach).

Using NCD for comparing the visual similarity of two web pages requires several implementation decisions: which format for rendered images; which compression algorithm; how to implement concatenation (i.e., by placing images side by side or one on top of the other). Furthermore, NCD is not symmetrical, i.e., $\mathrm{NCD}(x, y) <> \mathrm{NCD}(y, x)$, thus the ordering in the comparisons is not irrelevant. We computed $C(x)$ for an RGB image x with the following steps:

Step 1. We constructed r_x, that is, the in memory representation of x, in the form of array of bytes.

Step 2. We compressed r_x with LZMA algorithm obtaining the array of bytes c as output; we used the following configuration for LZMA: *preset* 9, the maximum compression; dictionary size 256 MiB; *raw* output format, *c* does not contain LZMA headers and checksums.

Step 3. $C(x)$ is the length of the byte-array *c*.

We computed $C(xy)$ for the concatenation of images *x* and *y* with the same steps, where the concatenation is obtained by stacking *x* on top of *y* (i.e., a vertical concatenation).

3.2. NCD between phishing page and targeted page

We considered two sets of screenshots: OP (Official Pages) and PP (Phishing Pages). We computed two sets of NCD distances. Let *p* denote a page in PP; we denote by $o(p)$ a page in OP among the ones for the organization targeted by *p*.

- ***S* (Same Company)**: for each page *p* in PP, NCD(p, $o(p)$);
- ***D* (Different Company)**: for each *p* in PP, for each page p' in OP such that $p' <> o(p)$, NCD(p, p');

In order to construct the above-mentioned sets, we considered only pairs of images with the same resolution; in case the same resolution is not available for both images, we did not compute the corresponding NCD.

Having constructed sets *S* and *D*, we compared the distributions of the NCD values in those sets for testing the following alternative hypotheses:

- H_0: The median of NCD values in *S* is equal to the median of NCD values in *D*;
- H_1: The median of NCD values in *S* is smaller than the median of NCD values in *D*;

If we accept H_1 then phishing pages tend to be indeed closer, in NCD terms, to the genuine page being targeted than to other genuine pages. This fact may be important as a foundation for visual phishing detectors.

We tested these hypotheses on the full sets *S*, *D* as well as on their subsets containing only values obtained from images with the same resolution. Such a distinction allows gaining deeper insights into whether phishing pages are actually designed to take the possibility of different browser resolutions into account or not. The results are given in Table 2. Each row corresponds to a test. Columns “Resolution” and “Browser” indicate the image pairs used in the test, while columns $|S|$ and $|D|$ contain, respectively, the number of pairs in *S* and the number of pairs in *D* used for that test.

It can be seen that our data indeed confirm H_1 (low *p*-values) for the full sets *S*, *D* as well as for nearly all their subsets that we have considered. We have emphasized in bold the only five cases in which *p*-value>0.05 and thus the data do not suffice to confirm H_1.

Our dataset thus supports the hypothesis that a phishing page is closer, in terms of NCD, to the page of the targeted organization than to pages of other organizations. This property could be useful as a foundation of a visual phishing detector as described in the introduction: The detector would compare the URL of the rendered page to the URL of the closest page in the set of pages to protect and raise a warning in case of a mismatch. Unfortunately, as we will show in the next section, NCD does not capture the notion of visual similarity adequately for a phishing detector.

Table 2. Statistical significance (*p*-value) of Wilcoxon test and *Z*-test (see the text)

Resolution	Browser	|*S*|	|*D*|	*p*-value Wilcoxon	*p*-value *Z*-test
All	All	45190	1112760	0.000	0.000
All	desktop	22595	556380	0.000	0.000
All	mobile	22595	556380	0.000	0.000
[1366, 768]	desktop	1509	37141	0.006	0.000
[1920, 1080]	desktop	1511	37189	0.007	0.000
[1440, 900]	desktop	1509	37141	**0.138**	0.000
[1600, 900]	desktop	1509	37141	**0.529**	0.000
[1280, 800]	desktop	1507	37093	0.017	0.000
[1536, 864]	desktop	1507	37093	**0.100**	0.000
[1280, 1024]	desktop	1509	37141	0.001	0.000
[1024, 768]	desktop	1505	37045	0.001	0.000
[1280, 720]	desktop	1505	37045	0.004	0.000
[1680, 1050]	desktop	1507	37093	**0.426**	0.000
[1360, 768]	desktop	1505	37045	0.000	0.000
[2560, 1440]	desktop	1509	37141	0.023	0.000
[1280, 768]	desktop	1505	37045	0.003	0.000
[1093, 615]	desktop	1505	37045	0.004	0.000
[1024, 600]	desktop	1503	36997	0.001	0.000
[768, 1024]	mobile	1505	36995	0.000	0.000
[1280, 800]	mobile	1508	37192	0.000	0.000
[600, 1024]	mobile	1510	37240	0.000	0.000
[601, 962]	mobile	1505	36945	0.000	0.000
[800, 1280]	mobile	1507	37093	0.000	0.000
[1024, 1366]	mobile	1509	37141	0.000	0.000
[360, 640]	mobile	1517	37333	0.000	0.000
[375, 667]	mobile	1496	37054	0.000	0.000
[720, 1280]	mobile	1516	37384	0.000	0.000
[414, 736]	mobile	1516	37334	0.000	0.000
[320, 568]	mobile	1514	37286	0.000	0.000
[320, 534]	mobile	1499	36951	0.000	0.000
[320, 570]	mobile	1494	36756	0.000	0.000
[480, 800]	mobile	1492	36758	0.000	0.000
[1080, 1920]	mobile	1497	36903	**0.051**	0.000

The reason is because pages that are very different in terms of NCD may nevertheless be visually very similar to each other. In other words, a phishing page may appear very similar to the targeted page although the two pages are associated with an NCD sufficiently large to not trigger the detector.

4. NCD and visual similarity

We constructed 773 pairs of screenshots taken from a phishing page *p* and from the corresponding targeted page *o*(*p*). Then, we visually assessed their differences in relation to their NCD distance. To obtain these pairs, for each phishing page *p* we considered only the *o*(*p*) with smallest NCD from *p*; and, we considered only the 1366×768 resolution, desktop mode. We emphasize that this is a qualitative assessment performed by a single subject and that this assessment is not sufficient to determine whether a given user will indeed fall prey of a phishing attack based on *p*. Despite this limitation, we believe our analysis may provide useful and interesting insights.

The most important findings are as follows:

- For pairs with NCD < 0.8100, all phishing pages look extremely similar, if not visually identical, to the targeted page.
- For pairs with NCD < 0.9957, only 45% of the pairs are visually similar.
- For pairs with 0.9957 < NCD, 98% of screenshot pairs are visually different. The remaining 2% of pairs are visually very similar.

The last finding is quite significant because it demonstrates that NCD may fail at detecting whether two pages are visually very similar. Unfortunately, as we will show in the next section, an attacker may exploit this fact rather easily, i.e., by crafting a phishing page *p* that looks very similar to the targeted page *o*(*p*) while ensuring that NCD(*p*, *o*(*p*)) is sufficiently high to fool a NCD-based phishing detector.

4.1. NCD distribution in our dataset

In order to gain deeper insights into the distribution of NCD values in our dataset, we proceeded as follows. We labelled each pair as either a GoodClone or a NotGoodClone, depending on the perceived similarity between the two elements of the pair. Then, we assessed the distribution of such a label based on the NCD distance between the two elements of the pair. To this end:

1. We computed the NCD distance corresponding to the first quartile NCD_1, the second quartile (median) NCD_2, the third quartile NCD_3 of the distribution of NCD values in set S (defined in the previous section). The results were 0.9993, 1.0307, 1.0560, respectively.
2. We split the interval of observed NCD values in four intervals: From 0 to NCD_1, from NCD_1 to NCD_2, from NCD_2 to NCD_3, from NCD_3 to the maximum NCD value observed. We named these intervals as Low, Mid-low, Mid-high and High, respectively.
3. We counted the percentage of GoodClone labels in each interval.

The results are in Table 3.

Table 3. Distribution of NCD values

NCD range	Interval name	GoodClone labels
NCD < 0.9957	Low	45%
0.9957<NCD<1.0307	Mid-low	3%
1.0307<NCD<1.0560	Mid-high	3%
1.0560 < NCD	High	1%

We provide interesting examples for the High interval below. In Figs 1-4, the images in each pair indeed look quite different visually (phishing page top, targeted page bottom). Despite the visual difference, though, the overall look of the phishing page could be perceived by users as sufficiently similar to that of the targeted page and thus suffice to execute an attack successfully. In other words, in these examples, the phishing page is sufficiently different from the targeted page to circumvent detection by a NCD-based visual phishing detector, yet the user might perceive the phishing page as being the legitimate one.

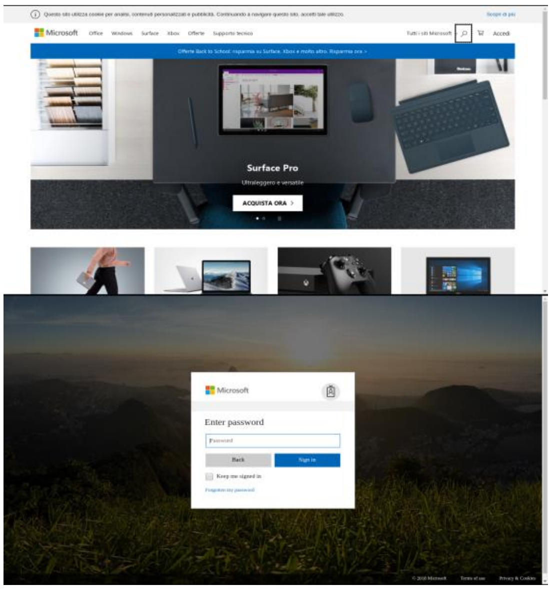


Fig. 1. High NCD difference (NCD=1.1100)

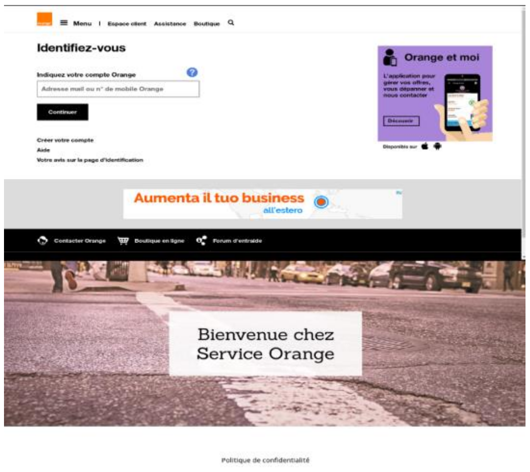

Menu
Espace client
Assistance
Boutique
Identifiez-vous
Indiquez votre compte Orange
Adresse mail ou n° de mobile Orange
Continuer
Créer votre compte
Aide
Votre avis sur la page d'identification
Orange et moi
Découvrir
Aumenta il tuo business all'estero
Contacter Orange
Boutique en ligne
Forum d'entraide
Bienvenue chez Service Orange
Politique de confidentialité

Fig. 2. High NCD difference (NCD=1.1100)

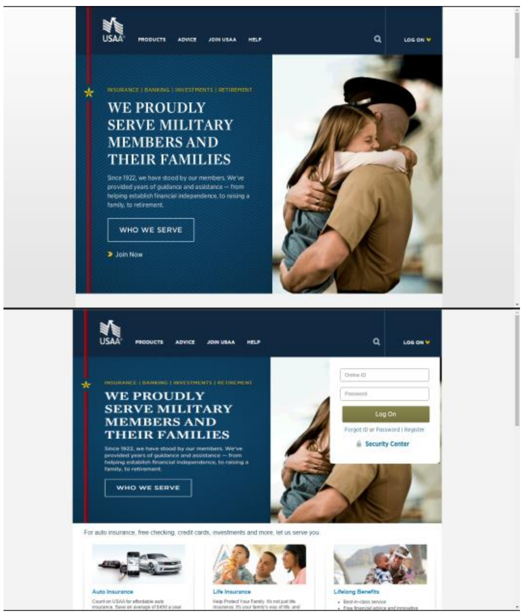


Fig. 3. High NCD difference (NCD=1.0800)

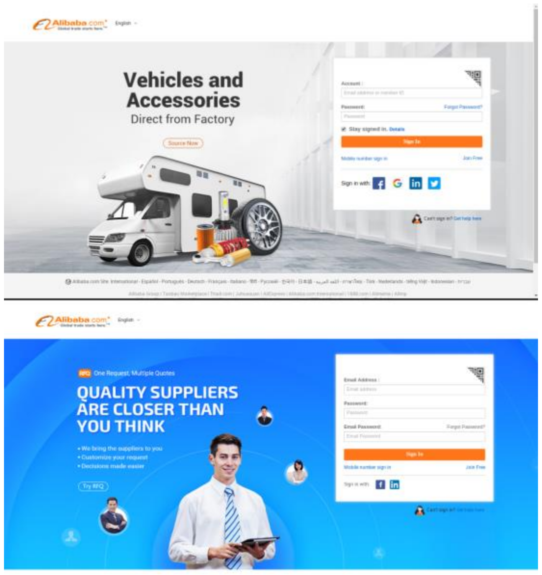


Fig. 4. High NCD difference (NCD=1.0600)

While the previous examples showed image pairs in which the phishing page is visually different from the targeted page but exhibits a very similar look, Figs 5 and 6 show image pairs in which the phishing page is nearly identical to the targeted page. In these figures, thus, detecting the phishing attack is even harder than in the previous case. Figs 5 and 6 are in the High interval (thus images in each pair are sufficiently different in terms of their NCD to not be detected by a NCD-based visual phishing detector) because the phishing page contains large images absent from the targeted page. This fact implies that the information conveyed by the two images in each pair is sufficiently different to lead to a large NCD value. Fig. 7 shows an image pair in which the phishing page is broken, thus it does not constitute any risk to users. Not surprisingly, this image pair is the High interval as well.

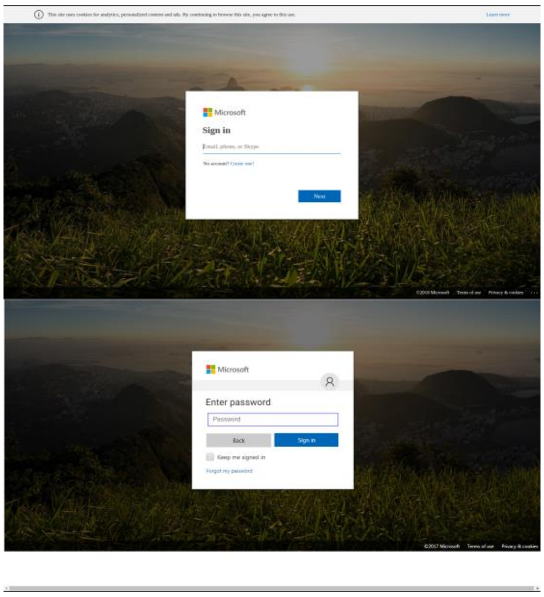


Fig. 5. High NCD difference (NCD=1.1000)

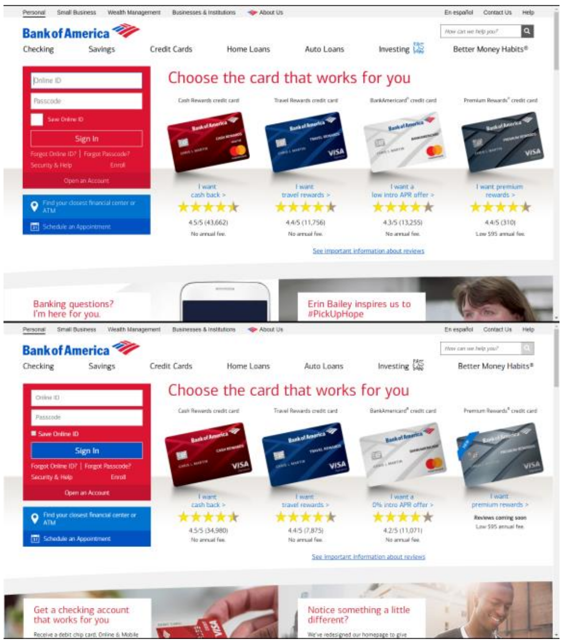
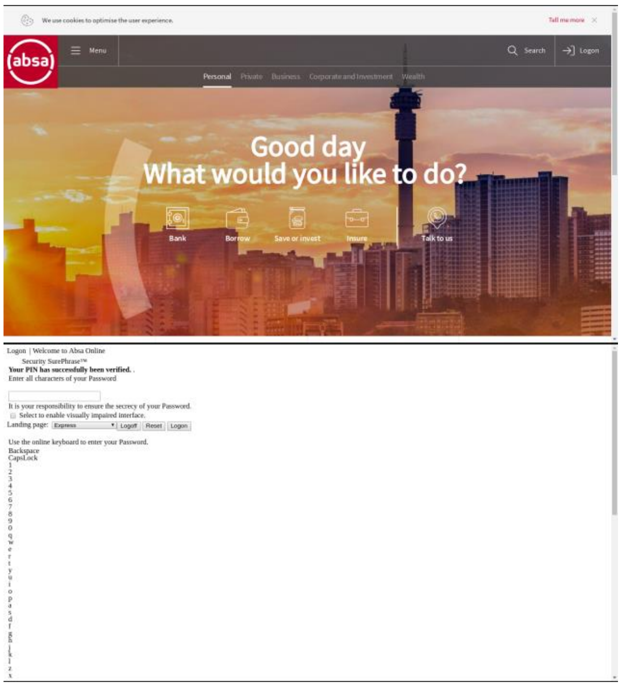


Fig. 6. High NCD difference (NCD=1.0700)



Logon | Welcome to Absa Online
Security SurePhrase™
Your PIN has successfully been verified. .
Enter all characters of your Password

It is your responsibility to ensure the secrecy of your Password.
Select to enable visually impaired interface.
Landing page: Express Logoff Reset Logon

Use the online keyboard to enter your Password.
Backspace
CapsLock
1
2
3
4
5
6
7
8
9
0
q
w
e
r
t
y
u
i
o
p
a
s
d
f
g
h
j
k
l
z
x

Fig. 7. High NCD difference (NCD=1.0700)

4.2. Deceiving NCD

While performing our analysis, we realized rather quickly that crafting a phishing page *p* that looks almost identical to the targeted page *o*(*p*), while ensuring that NCD(*p*, *o*(*p*)) is large (thereby circumventing a visual phishing detector based on NCD) is quite simple. All that is needed is applying a transformation to *o*(*p*) that changes its visual appearance only slightly, while leading to a very different bit level representation. Merely altering brightness and/or zoom of *o*(*p*) is a simple and effective means of implementing such transformation. Table 4 quantifies the NCD between a phishing page and the targeted page when the former is constructed by applying these transformations to the latter. Fig. 8 shows the original page and the page obtained with both transformations considered.

Table 4. NCD distance of a phishing page obtained with adversarial transformations to the targeted page from the targeted page itself

Transformation	NCD
Brightness increased by 5%	0.99676
Centered zoom-in of 5% performed with Bilinear interpolation	0.8293159
Both (brightness and then zoom)	0.9957

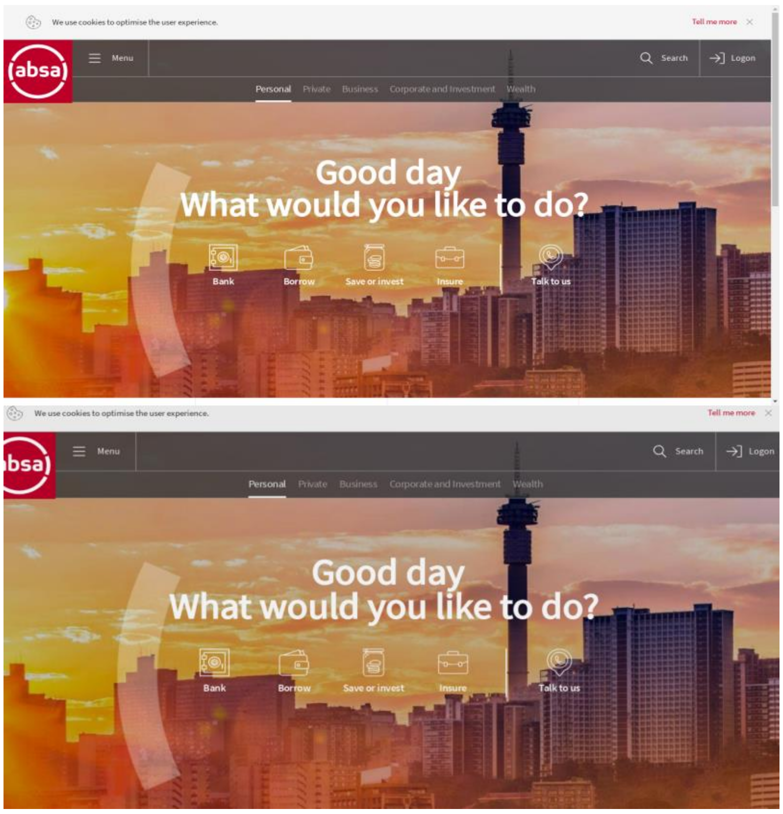


Fig. 8. Original page (top), phishing page with high NCD obtained by altering brightness and zoom only slightly (bottom)

To place these figures in perspective, it suffices to recall the findings in the previous section, which show that a phishing detector calibrated for classifying a page as suspicious with NCD values above 0.9957 would flood the user of meaningless warnings (false positives), thereby making the approach impractical. Specifically, by using the value in the table as a classification threshold (0.9957), our dataset would exhibit 55% false positive rate.

We did not apply these transformations systematically to our full dataset; however, we verified on several randomly selected images that this finding appear to have general validity. Furthermore, we remark that we considered only two very simple transformations that do not depend on the targeted page. In the page of the figure an attacker would have many other opportunities for increasing NCD further while maintaining visual similarity, for example by slightly altering the color of the logo, of the banner, of the fonts. In general, an even more effective strategy consists in modifying the targeted page by adding elements that are small (so that they do not raise suspicions to the user) and complex (so that their bit representation cannot be compressed very much, thereby contributing to a larger NCD distance from the targeted page).

5. Conclusion

There is an urgent need of developing phishing detection mechanisms able to complement current defenses based on the reputation of the page accessed by the browser. Such mechanisms should be close to browsers and based solely on the visual appearance of the rendered page. A crucial prerequisite for developing such mechanisms consists in the availability of a large and diverse collection of screenshots of phishing pages. Unfortunately, publicly releasing such a dataset would violate the terms of service of targeted companies and collecting such a dataset raises several technical challenges.

We have described such challenges in detail along with practical solutions. The proposed procedure allowed us to collect tens of thousands of screenshots, from hundreds of different phishing pages targeting about 20 different companies. Based on the collected dataset, we could analyze the appearance of real phishing pages at different resolutions. We found that phishing pages tend to render correctly at all resolutions used in desktops but tend to have more problems at resolutions used in tablets and smartphones. Most importantly, this dataset allowed us to assess in detail the usage of a similarity metric based on information theory that could constitute a powerful foundation for visual phishing detectors.

Unfortunately, our analysis has shown that pages that are very different according to this metric may still be perceived as “sufficiently similar” by a human. Furthermore, we have also shown that there are simple ways for an attacker to craft a phishing page that looks very similar to the targeted page but is very dissimilar to that page according to the considered metric. This fact implies that a visual phishing detector based on that similarity metric is unlikely to work. The key issue is that even if the phishing page and the targeted page convey highly different information, such as different background images, such a difference may not be perceived as such by a user. According to our analysis, a visual phishing detector should not focus on the detection in a page of portions that are identical to portions of pages to be protected, but rather it should detect more high-level similarities in the overall style and look, which is still a difficult research issue.

References

1. The Human Factor: People-Centered Threats Define the Landscape [Internet]. Proofpoint, 2018. **https://www.proofpoint.com/us/human-factor-2018**
2. L a z a r, L. Our Analysis of 1,019 Phishing Kits – Blog | Imperva. – In: Blog | Imperva [Internet]. 4 January 2018 [cited 4 July 2018]. **https://www.imperva.com/blog/2018/01/our-analysis-of-1019-phishing-kits/**
3. M a u r e r, M.-E., D. H e r z n e r. Using Visual Website Similarity for Phishing Detection and Reporting. – CHI’12 Extended Abstracts on Human Factors in Computing Systems, New York, NY, USA, ACM, 2012, pp. 1625-1630.
4. A f r o z, S., R. G r e e n s t a d t. PhishZoo: Detecting Phishing Websites by Looking at Them. – In: 2011 IEEE 5th International Conference on Semantic Computing, 2011, pp. 368-375.
5. C h e n, T.-C., S. D i c k, J. M i l l e r. Detecting Visually Similar Web Pages: Application to Phishing Detection. – ACM Trans. Internet Technol., New York, NY, USA, ACM, Vol. **10**, 2010, pp. 5:1-5:38.

6. M e d v e t, E., E. K i r d a, C. K r u e g e l. Visual-Similarity-Based Phishing Detection. – In: Proc. of 4th International Conference on Security and Privacy in Communication Netowrks, New York, NY, USA, ACM, 2008, pp. 22:1-22:6.
7. L o w e, D. G. Distinctive Image Features from Scale-Invariant Keypoints. – Int. J. Comput. Vis., Vol. **60**, 2004, pp. 91-110.
8. B a y, H., A. E s s, T. T u y t e l a a r s, L. V a n G o o l. Speeded-Up Robust Features (SURF). – Comput Vis Image Underst., Vol. **110**, 2008, pp. 346-359.
9. M e d v e t, E., A. B a r t o l i, G. D a v a n z o, A. D. L o r e n z o. Automatic Face Annotation in News Images by Mining the Web. – In: Proc. of 2011 IEEE/WIC/ACM International Conferences on Web Intelligence and Intelligent Agent Technology, 2011, pp. 47-54.
10. D e L o r e n z o, A., E. M e d v e t, F. T a r l a o, A. B a r t o l i. Personalized, Browser-Based Visual Phishing Detection Based on Deep Learning. – In: Proc. of 13th International Conference on Risks and Security of Internet and Systems (CRiSIS).
11. C h e n, T.-C., T. S t e p a n, S. D i c k, J. M i l l e r. An Anti-Phishing System Employing Diffused Information. – ACM Trans. Inf. Syst. Secur., New York, NY, USA, ACM, Vol. **16**, 2014, pp. 16:1-16:31.
12. C i l i b r a s i, R., P. M. B. V i t a n y i. Clustering by Compression. – IEEE Trans. Inf. Theory, Piscataway, NJ, USA, IEEE Press, Vol. **51**, 2005, pp. 1523-1545.
13. V á z q u e z, P.-P., J. M a r c o. Using Normalized Compression Distance for Image Similarity Measurement: An Experimental Study. – Vis. Comput., Vol. **28**, 2012, pp. 1063-1084.
14. A x e l s s o n, S. Using Normalized Compression Distance for Classifying File Fragments. – In: Proc. of 2010 International Conference on Availability, Reliability and Security, 2010, pp. 641-646.
15. T e l l e s, G. P., R. M i n g h i m, F. V. P a u l o v i c h. Normalized Compression Distance for Visual Analysis of Document Collections. – Comput. Graph., Vol. **31**, 2007, pp. 327-337.
16. B a r t o l i, A., G. D a v a n z o, E. M e d v e t. The Reaction Time to Web Site Defacements. – IEEE Internet Comput, Vol. **13**, 2009, pp. 52-58.
ieeexplore.ieee.org

Appendix. Life time of a phishing page

For completeness of analysis, we attempted to estimate the *life time* of a phishing page, i.e., for how long the page is made available by the server on which the page is hosted. Usage of Phishtank data unavoidably leads to a raw estimation of such a figure, for several reasons. In particular, when a page is inserted in Phishtank, the page may have been active already for an unknown time; furthermore, Phishtank updates its dataset once every hour.

The life time of a phishing page is usually longer than the time for which the phishing page may be accessed by all potential victims, because many phishing pages are blocked by the browser, or by some defensive mechanism along the path between the browser and the web server hosting the phishing page. Indeed, the presence of a page in Phishtank is often the event that triggers such mechanisms. However, estimating the life time of a phishing page is important in order to gain insights into the *reaction time* of the infrastructure hosting the page. Pages that are blocked by most browsers and web firewalls should not remain active for a long time: The infrastructure should deactivate the account responsible for the page, or take similar actions, quickly. In this respect, we remark that phishing attacks are increasingly using each page in fewer email messages in order to bypass the most effective and most diffused security mechanisms, i.e., those relying on the reputation of the page being accessed. Thus, obtaining information on the typical reaction time is important to better understand the current scenario.

We followed a procedure similar to the one used in [16] (the cited paper estimated the reaction time to web defacements based on the content of Zone-H, a web archive that played for defacements the same role that Phishtank plays for phishing pages): we downloaded the Phishtank database and checked the availability of each URL every hour; we assumed that when a download returns an error, either at the TCP level, or at the TLS level, or at the HTTP level, the page is no longer available; we considered the time interval between the *download error* and the *submission time* (an information that Phishtank makes available for each page in its database) as an approximation of the life time of that page.

We executed this procedure with the same settings described in Section 2.1, i.e., we disabled the integrated warnings for dangerous and deceptive web pages in the headless Google Chrome browser, and conveyed all the traffic through the TOR anonymity network. Since the Phishtank database contains many entries associated with "old" phishing pages that are no longer available, we considered only entries that were inserted less than 2 days since the beginning of our monitoring activity. Furthermore, in order to keep the number of URLs to check more manageable, we stopped checking each page that has been available for at least two days. As a result of the choices just described, we could check 42,155 URLs. We omit the estimated cumulative probability distribution for space reasons and provide only the corresponding values for the first quartile, second quartile (median) and third quartile: 1 hour, 8 hour and 37 hours respectively. These figures demonstrate that the administrators of the infrastructures that host phishing pages tend to react quite slowly despite the fact that those pages have been inserted in a public, easily accessible and widely known archive of phishing attacks.

Received 30.10.2018; Accepted 19.11.2018