Regular Paper

Detection Method of Homograph Internationalized Domain Names with OCR

Yuta Sawabe[1,†1,a)] Daiki Chiba[2,b)] Mitsuaki Akiyama[2] Shigeki Goto[1]

Received: November 27, 2018, Accepted: March 5, 2019

Abstract: Currently, many attacks are targeting legitimate domain names. In homograph attacks, attackers exploit human visual misrecognition, thereby leading users to visit different (fake) sites. These attacks involve the generation of new domain names that appear similar to an existing legitimate domain name by replacing several characters in the legitimate name with others that are visually similar. Specifically, internationalized domain names (IDNs), which may contain non-ASCII characters, can be used to generate/register many similar IDNs (homograph IDNs) for their application as phishing sites. A conventional method of detecting such homograph IDNs uses a predefined mapping between ASCII and similar non-ASCII characters. However, this approach has two major limitations: (1) it cannot detect homograph IDNs comprising characters that are not defined in the mapping and (2) the mapping must be manually updated. Herein, we propose a new method for detecting homograph IDNs using optical character recognition (OCR). By focusing on the idea that homograph IDNs are visually similar to legitimate domain names, we leverage OCR techniques to recognize such similarities automatically. Further, we compare our approach with a conventional method in evaluations employing 3.19 million real (registered) and 10,000 malicious IDNs. Results reveal that our method can automatically detect homograph IDNs that cannot be detected when using the conventional approach.

Keywords: domain name, IDN, homograph attack, OCR

1. Introduction

On the Internet, domain names are used in almost all of the world's websites and e-mail addresses. They were originally introduced to convert numeric Internet Protocol (IP) addresses into human-readable strings and generally include the name of the corresponding service or company. However, cyber attackers often exploit this by registering domain names that are similar to those employed for legitimate services. Such attacks can broadly be classified into two types. The first type is called typosquatting [2]. In this type, attackers exploit human typing errors by generating and registering domain names that are similar to legitimate ones, but several characters are replaced (or inserted) with the ones that are close on the user's keyboard. The other type is called homograph attack [3]. In this type, attackers exploit mistakes in human visual recognition by generating and registering domain names wherein some characters have been replaced by visually similar characters. We call these domain names *homograph domain names*. Since the introduction of internationalized domain names (IDNs), Unicode characters can be used in domain names. In comparison with domain names generated via typosquatting, this development allows homograph attacks to generate more diverse domain names that are similar to real domain names. Attackers have already started to generate malicious homograph domain names by employing IDNs for use as malicious (e.g., phishing) websites [4]. We call these domain names *homograph IDNs*.

The conventional approach of detecting such homograph IDNs involves the use of a predefined mapping between characters that are visually similar [5], [6]. The mapping tables list pairs of non-ASCII and visually similar ASCII characters. These tables are used to convert the non-ASCII characters in IDNs into ASCII ones and detect homograph IDNs. However, this approach has two major limitations. First, it cannot detect homograph IDNs with characters that are not defined in the mapping. Second, the mapping must be manually updated. In this study, we propose a new method to address these drawbacks by detecting homograph IDNs via dynamic (rather than pre-defined) mapping between similar characters. By employing target IDNs and a list of legitimate/popular domain names as the input, the target IDNs are converted into images and optical character recognition (OCR) is employed to detect whether the IDNs are similar to legitimate domain names. This approach aims to take advantage of the main fact that homograph IDNs are visually similar to legitimate domain names. We leverage OCR techniques to recognize such similarities automatically and extract homograph IDNs and their corresponding legitimate domain names. The contributions of our study are as follows:

- To the best of our knowledge, the proposed method is the first to use OCR to automatically detect homograph IDNs.
- We evaluate the effectiveness of our method using 3.19 mil-

[1] Waseda University, Shinjuku, Tokyo 169–8555, Japan
[2] NTT Secure Platform Laboratories, Musashino, Tokyo 180–8585, Japan
[†1] Presently with NTT Security (Japan) KK
[a)] sawabe.yuta@ruri.waseda.jp
[b)] daiki.chiba@ieee.org

This paper is an extended version of a paper published in APAN Research Workshop 2018 [1].

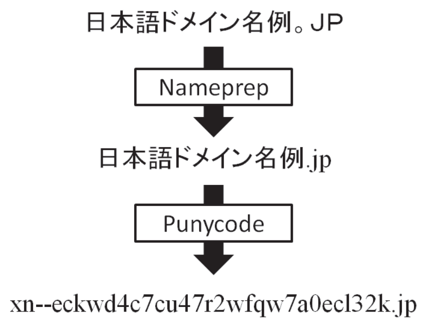


Fig. 1 Example of IDNA conversion.

lion real (registered) and 10,000 malicious IDNs.

- Our method can dynamically develop a new mapping comprising non-ASCII characters that cannot be recognized using a conventional method.

2. Background

2.1 Internationalized Domain Names

Initially, domain names may only include ASCII letters, numbers, and dashes because the Internet is originally designed for the English language. However, as the Internet is more extensively used by non-English-speaking countries and because multiple languages can be used for websites and emails, demands on the use of multiple languages for domain names are also increasing. To address this issue, IDNs were invented to enable the use of non-ASCII characters in domain names [7]. Recently, it has also become possible to use non-ASCII characters in top-level domain (TLD) names known as IDN TLDs. IDNs are implemented in a manner that they remain compatible with existing domain names and domain name servers. Specifically, there is a mechanism called Internationalized Domain Names in Applications (IDNA) for converting IDNs into ASCII strings. This mechanism is defined in RFC 3490 [8]. **Figure 1** shows an example of a Japanese IDN being converted into an ASCII string by employing IDNA. IDNA involves two processes, namely, Nameprep and Punycode.

Nameprep, which is defined in RFC 3491 [9], preprocesses the domain names to eliminate notation variations and is based on the Stringprep profile defined in RFC 3454 [10]. It involves three steps as follows: mapping, normalization, and prohibited output.

First, the mapping step unifies the character types and converts all uppercase letters into lowercase ones. Then, normalization handles any special characters by converting half-width characters into full-width characters, splitting composite characters, and converting superscripts and subscripts into ordinary characters. Finally, the prohibited output step returns an error if the domain name contains forbidden characters (e.g., blank or control characters).

Punycode is an encoding syntax defined in RFC 3492 [11]. The encoding is reversible, and translated strings are prefixed with "`xn--`" to distinguish them from normal domain names. The encoding process is as follows:

(1) Remove all non-ASCII characters from the domain name and append "-" at the end of the remaining string.
(2) Calculate the numerical character code and original position for each removed non-ASCII character, and replace them with corresponding strings.
(3) Append the converted strings at the end of the original string.

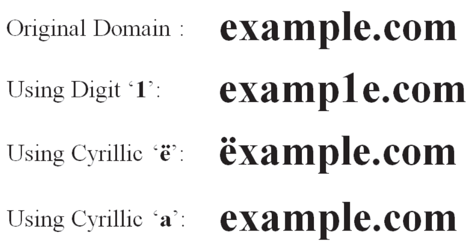

Original Domain : example.com
Using Digit '1': examp1e.com
Using Cyrillic 'ё': ёxample.com
Using Cyrillic 'а': exаmple.com

Fig. 2 Example of homograph domain names.

2.2 Homograph Attacks

In a homograph attack, an attacker generates and registers domain names by replacing several characters of legitimate/popular domain names with other strings. **Figure 2** illustrates several examples of homograph domain names for `example.com`. The second domain name in the figure is a homograph that has been generated by replacing letter "`l`" in the original domain with digit "`1`." Users who read this domain name will possibly think that this name is legitimate based on its appearance, but they may end up visiting the fake domain. The third and fourth domains in the figure are homograph IDNs that are created by replacing the characters in the original domain name with Cyrillic characters. Many non-ASCII characters with shapes similar to those of ASCII characters are available. Hence, these characters can be used to generate multiple domain names that cannot be easily and immediately distinguished from the real ones. In this manner, homograph attacks render users to recognize similar but fake domain names as legitimate ones, thereby directing them to unexpected sites.

Homograph IDNs have been already used in phishing attacks [4]. Thus, several web browsers currently employ countermeasures that prevent homograph attacks by displaying the domain names in Punycode format when they encounter characters in different languages [12]. However, Xudong and Wordfence have developed homograph IDNs that can avoid such countermeasures; therefore, users remain at risk [13], [14]. These homographs use a specific trick: they replace all characters in legitimate domain names with letters from another language to prevent web browsers from identifying the characters as homographs. Several holders who manage legitimate domain names register homograph domain names for brand protection, but only a few take such countermeasures because registering and managing domain names are costly [15]. As a result, although we are already aware of the threats of homograph attacks, we still need sufficient countermeasures against these attacks.

3. Proposed method

This section describes the proposed method for detecting homograph IDNs. First, Section 3.1 provides an overview of the method, whereas Sections 3.2–3.5 describe the followed steps in detail.

3.1 Overview

In this study, we propose the use of OCR to detect homograph IDNs. The proposed method uses the target IDNs and a list of

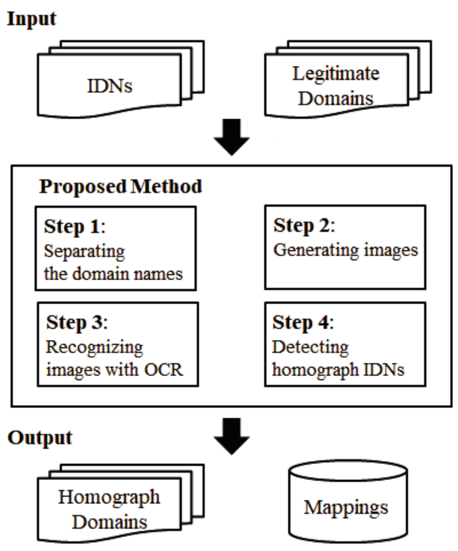

Input
IDNs
Legitimate Domains
Proposed Method
Step 1: Separating the domain names
Step 2: Generating images
Step 3: Recognizing images with OCR
Step 4: Detecting homograph IDNs
Output
Homograph Domains
Mappings

Fig. 3 Overview of the proposed method.

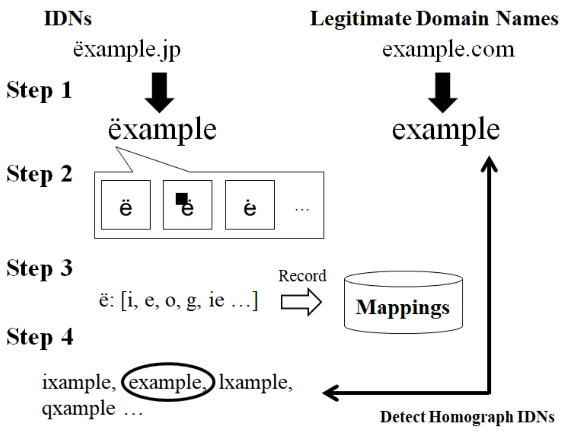


Fig. 4 Example of generating candidate legitimate domain names using the proposed method.

legitimate/popular domain names as inputs and produces a list of detected homograph IDNs as well as their corresponding legitimate sites. The method also outputs a new mapping between non-ASCII characters and similar ASCII characters based on the OCR results. **Figure 3** shows an overview of the proposed method, which comprises four steps: separating the domain names, generating images, recognizing the images using OCR, and detecting homograph IDNs. These steps are explained in detail as follows. **Figure 4** shows an example of the generation of candidate legitimate domain names from a homograph IDN using the proposed method.

3.2 Step 1: Separating the domain names

This step removes the public suffixes from both the input IDNs and legitimate domain names. Public suffixes are strings in domain names that cannot be controlled by individual users [16], such as generic TLDs (gTLDs) (e.g., `.com` or `.net`) or country code TLDs (ccTLDs) (e.g., `.co.jp` or `.co.uk`). We include this step because attackers do not necessarily use the same public suffixes of the target legitimate domains for their homograph IDNs. For example, attackers generating homograph IDNs targeting `example.com` might generate not only `ëxample.com` but also `ëxample.co.jp`. Specifically, in October 2013, many new gTLDs, such as `.xyz` or `.top`, were introduced. Consequently, the number of available public suffixes is rapidly increasing and these new gTLDs are known to have been used in attacks [17]. Hence, this step enables us to expand the range of detectable homograph IDNs. Then, the remaining domain name strings are split into string arrays based on their dot (“`.`”) characters.

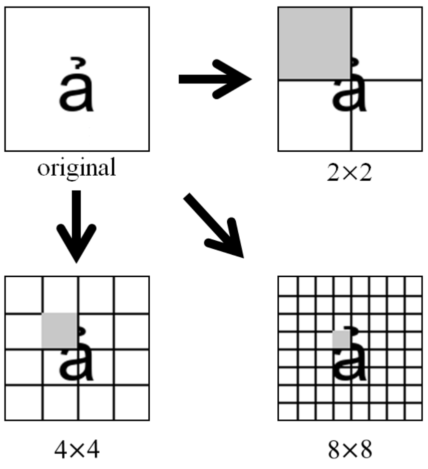


Fig. 5 Example of white masks.

3.3 Step 2: Generating images

First, we convert the strings generated in Step 1 into images to be used with OCR. OCR systems can usually extract individual characters, but their recognition accuracy is low for small (e.g., “`.`”) and linear (e.g., “`i`” and “`l`”) characters. To avoid any decrease in accuracy, we generate a sequence of single-character images from domain name strings.

Then, we perform some preliminary image processing to accurately recognize various homograph IDNs with OCR. Using OCR, our aim is to automatically recognize the Unicode characters used by attackers in homograph IDNs to resemble ASCII characters. For example, in Fig. 4, we require the OCR system to read letter “ë” and convert it into letter “e.” However, a high-performance OCR system may identify “ë” and “e” as different in shape, thereby producing unexpected results. To obtain suitable and varied results, we deliberately create images where part of the image has been blanked out to alter the character shape, thus influencing the recognition results. We call these images *mask images*. We use two fill colors (i.e., black and white) for these images to add noise and remove part of the character, respectively. In addition, three different sizes of masks were constructed by dividing the images into 2×2, 4×4, or 8×8 square grids and filling one square in the images with black or white. Hence, we generate a total of 169 images for each character: 168 mask images (using 84 different mask areas and two colors, i.e., black and white) and the original unmasked image. **Figure 5** shows an example of image masking with different sizes of white masks. For clarity, the mask color is presented as gray rather than white in the figure and lines have been added to mark the squares.

3.4 Step 3: Recognizing images with OCR

We apply OCR to the images generated in Step 2 to convert the characters in the domain name into visually similar ones. This step may produce various results for images of the same character depending on the color and size of the masks employed; thus, multiple results are recorded for each character. This step can be simplified by reusing these results for repeated characters. Our method can dynamically generate these mappings based on the input IDNs without requiring them to be prepared beforehand.

3.5 Step 4: Detecting homograph IDNs

Here, we assume that the input IDNs are homographs and use the mappings found in Step 3 to generate their legitimate candidate domain names. We consider all possible combinations for IDNs that contain multiple Unicode characters. If these candidates involve one of the input legitimate domains, then the corresponding input IDN is detected as a homograph and is output with the associated legitimate domain.

If all results are recorded in Step 3, then the number of generated candidates might be enormous. This action also increases the possibility of false positives because the results that are not visually similar to the character are included. Meanwhile, if the number of registered results is very small, detecting various homograph IDNs is impossible. Thus, preventing erroneous detection is necessary. We described this adjustment method in detail in Section 4.4.2.

4. Evaluation

In this section, we evaluate the effectiveness of the proposed method described in Section 3. Specifically, we compare both the number and characteristics of the homograph IDNs detected by applying the proposed and conventional methods to two real-world datasets.

4.1 Performance Comparison

The conventional method of detecting homograph IDNs uses a predefined mapping: a list of non-ASCII characters and visually similar ASCII characters. In this evaluation, we combined two existing mappings to establish a baseline method for comparison.

The first mapping was defined and used in the dnstwist script [5], which is a tool for finding malicious domain names that can be used by attackers. When a domain name is input to dnstwist, it generates candidate malicious domain names, including homographs, and presents DNS results for all candidates. Dnstwist uses a mapping between all alphabetic characters and visually similar strings to generate homograph domain names. We use this mapping in reverse fashion to implement a mapping between non-ASCII and visually similar ASCII characters.

The other existing mapping is the table provided by the Unicode Consortium [6], indicating the relationships between similar characters. The table lists pairs of the Unicode characters that have a similar appearance. In this evaluation, we use version 11.0.0 of the table, which was developed in 2018. Furthermore, we only selected the character pairs wherein the corresponding character was an alphabet.

As a baseline method, we used these two mappings to convert non-ASCII characters in IDNs into visually similar ASCII ones. To compare the performance of this method with that of the proposed method, we devised an algorithm that uses the IDNs and legitimate domain names as inputs, employs the mappings to generate legitimate candidate domain names, and outputs the detected homograph IDNs.

Table 1 Datasets.

Dataset	Date	#IDNs
Project Sonar Dataset	2018-08-03–2018-08-04	3,198,144
Malicious Dataset	2017-10-29–2017-11-10	16,341

4.2 Datasets

We used two datasets for our evaluation. The first one was a forward DNS dataset provided by Project Sonar [18]. This dataset contains only those domain names that have A, AAAA, or ANY DNS record; therefore, all domains in the dataset are completely associated with IP addresses. From this dataset, we only extracted IDNs.

The other dataset comprises a list of malicious domain names obtained from public blacklists, such as hpHosts [19], MalwareDomains [20], and PhishTank [21], together with commercial blacklists, such as Spamhaus [22] and URIBL [23]. We extracted the IDNs from these blacklists. **Table 1** summarizes both datasets.

4.3 Method

We used the IDN datasets described in Section 4.2 to compare the number and characteristics of the homograph IDNs detected by the conventional and proposed methods by targeting homograph IDNs that satisfy the following two conditions:

(1) Non-ASCII characters that are visually similar to alphabet characters are used.
(2) The domain names of legitimate/popular sites with many users are targeted.

To satisfy the first condition, the OCR results were limited only to alphabetic characters. Furthermore, the similarity between ASCII characters were not considered in this evaluation; hence, we did not apply OCR to the images of ASCII characters. To satisfy the second condition, we collected the top 1,000 domain names ranked by Alexa Top Sites [24] on Oct. 1, 2017 and used them as legitimate domain names.

4.4 Initial Preparation

4.4.1 Legitimate Domain Name

First, we created a list of legitimate domain names for evaluation. This list was not as simple as directly using the Alexa Top Sites data because several domain names in the data only differed in public suffixes. For example, although `google.com` was in the first place, 24 Google domain names are in the top 100 that vary only in their public suffixes (e.g., `google.co.in` and `google.co.jp`) and 75 are in the top 1,000. Therefore, we removed all duplicate domain names except those with the highest rankings.

In addition, we considered the domain name length. For extremely short names (excluding the public suffix), the same string will be possibly erroneously generated among the candidate domain names, thereby affecting our evaluation. To avoid this issue,

Table 2 Results for the Project Sonar dataset.

Legitimate Domain Name	Alexa Rank	#Detected Homograph IDNs				Brand Protection
		Only Conventional Method	Both Conventional and Proposed Methods	Only Proposed Method	Total	
google	1	0	149	74	223	30
facebook	3	1	101	66	168	0
apple	58	1	67	86	155	0
icloud	368	1	28	69	98	0
amazon	10	2	69	27	98	32
bittrex	386	2	18	76	96	1
blockchain	884	2	40	32	74	0
instagram	17	7	25	28	60	0
yahoo	6	0	43	13	56	31
twitter	13	4	32	19	55	5
paypal	61	0	23	28	51	3
youtube	2	0	22	23	45	0
hotels	748	0	9	35	44	0
whatsapp	69	0	33	3	36	0
coinbase	357	2	8	23	33	0
skype	345	0	18	14	32	30
microsoft	47	1	17	10	28	0
wikipedia	5	1	24	3	28	0
steamcommunity	169	1	8	18	27	0
linkedin	30	2	11	9	22	0
Total		115 (5.3%)	1,118 (51.1%)	955 (43.6%)	2,188 (100.0%)	166 (7.6%)

Table 3 Results for the malicious dataset.

Legitimate Domain Name	Alexa Rank	#Detected Homograph IDNs			
		Only Conventional Method	Both Conventional and Proposed Methods	Only Proposed Method	Total
paypal	61	0	4	2	6
apple	50	0	4	0	4
facebook	3	0	2	1	3
icloud	368	0	2	1	3
google	1	0	2	0	2
steamcommunity	169	0	0	2	2
elpais	405	0	1	0	1
Total		0 (0.0%)	15 (71.4%)	6 (28.6%)	21 (100.0%)

we removed any legitimate domain names with four-character length or less. After executing both these steps, we still have 723 of the original 1,000 domain names to be used as legitimate domain names.

4.4.2 OCR setting

Herein, we used the open-source Tesseract OCR [25] package for Step 3 of our method. Using this package, we can specify the target language to be recognized and the characters to be allowed in the resulting text in advance. In this evaluation, only alphabetic characters were allowed.

As described in Section 3.3, we generated 169 images for each character. In our evaluation, we set the image size to 128 × 128 pixels. We investigated the relationship between the fonts used in Step 2 and the OCR recognition accuracy in advance. We found that the accuracy tends to decrease when the font weight is small, i.e., the line width of a character is thin. Thus, we use the Arial font in this evaluation because the weight of this font is not small and the Arial font has many characters. Although we recorded the results of applying OCR to all images, the number of candidate domain names would dramatically increase if we use all results to generate the possible mappings. Instead, only the top 10 most frequent results were utilized to implement the mappings.

Next, we described the method of preventing misdetection using OCR. In this evaluation, we used the confidence generated by the Tesseract OCR [26]. The confidence is a score that shows the accuracy of image readings. We reduced the false positive rate by employing the following procedure. First, we recorded all the confidence scores and the OCR results in Step 2. Then, we calculated the average of the confidence score corresponding to each character in the candidate domain name. Finally, if the IDN was detected as a homograph IDN in Step 4 and the calculated average score was below the threshold, then the IDN was identified as misdetection. We also investigated several non-ASCII characters and their reading results in advance and then set the confidence score threshold to 80.0. Moreover, as we apply OCR to the images of only non-ASCII characters in this evaluation, we calculate the ASCII character confidence score as 100.0.

5. Results

Table 2 lists the number of homograph IDNs detected by the conventional and proposed methods for the Project Sonar dataset. This table lists the number of homographs detected using the conventional method, both conventional and proposed methods, and proposed method alone for each legitimate domain name. Owing to the limited space of the page, this table only lists the top 20 legitimate/popular sites with the most detected homograph IDNs. Several non-homograph IDNs are detected by the proposed method; hence, we only count the homograph IDNs. Based on these results, the proposed method can detect a large number of homograph IDNs that were not detected by the conventional method. Specifically, the proposed method detected approximately 1.68 times more homograph IDNs than the conventional method. However, a few domain names were also detected using the conventional method but not by the proposed method.

Table 3 summarizes the results for the malicious IDN dataset. Considering that the number of detected homograph IDNs was small in this case, Table 3 lists the number of all legitimate sites involved. As can be observed from the table, some homograph IDNs were used for malicious purposes and the proposed method can detect malicious IDNs that were not detected by the conventional method. Section 6 comprehensively discusses several detected homographs.

By focusing on the legitimate domain names listed in Tables 2 and 3, we can observe two significant features. First, many domains that are highly ranked by Alexa Top Sites are included. Thus, the attackers have developed many homograph IDNs similar to legitimate/popular sites with high traffic, directing more victims to different sites. Second, all legitimate sites listed in these tables require user accounts. Homograph attacks can direct users to malicious sites that appear similar to legitimate ones. By focusing on sites with user accounts, attackers can steal personal information, including user IDs and passwords.

Table 4 lists the number of non-ASCII characters in the detected homograph IDNs. Based on this result, more than 70% of homograph IDNs are generated by replacing one character only. Meanwhile, several detected homograph IDNs use five or more non-ASCII characters. Generally, a total of 53 homographs are generated by replacing all the characters of the targeted legitimate domain names.

In addition, we investigate the relationship between the threshold of the confidence score described in Section 4.4.2 and the number of detected homograph IDNs. **Figure 6** shows the result of accuracy evaluation while the threshold is changed. If the threshold score is high, then the accuracy rate also increases because the candidates not similar to the legitimate domain are removed, causing a decrease in the number of detected homograph IDNs. Thus, setting an appropriate threshold score according to a purpose is necessary.

Finally, we present the survey results for brand protection. Table 2 also lists the number of homograph IDNs registered by legitimate owners for brand protection. We use the WHOIS Data collected on Aug. 16, 2018. As presented in the table, many legitimate holders do not register corresponding homograph IDNs for brand protection. Although several legitimate owners take countermeasures, the number of detected homograph IDNs in this evaluation is higher than the number of registered IDNs. **Table 5** presents the TLDs of the detected homograph IDNs. It separately describes the homograph IDNs that are registered for domain protection and then describes the ones that are not. Based on this result, legitimate domain owners register IDNs that use only TLDs, such as "`.com`" or "`.net`." Meanwhile, although most homograph IDNs registered by attackers also use "`.com`" domain names, they generate homograph IDNs with diverse TLDs (up to 76). Meanwhile, attackers register many homograph IDNs with various TLDs. In particular, they use new gTLDs, such as "`.top`" or "`.xyz`," for homograph attacks, but legitimate domain holders do not register such homograph IDNs. Hence, they were not able to take sufficient countermeasures for these homograph IDNs. In this situation, the proposed method is highly effective in detecting homograph IDNs from the user's perspective and preventing the users from accessing malicious sites.

Table 4 Number of non-ASCII characters in homograph IDNs.

Number of Non-ASCII Characters	#IDNs
1	1,532 (70.0%)
2	487 (22.3%)
3	109 (5.0%)
4	3 (0.1%)
5	24 (1.1%)
6	20 (0.9%)
7	5 (0.2%)
8	6 (0.3%)
9	0 (0%)
10	2 (0.1%)
Total	2,188

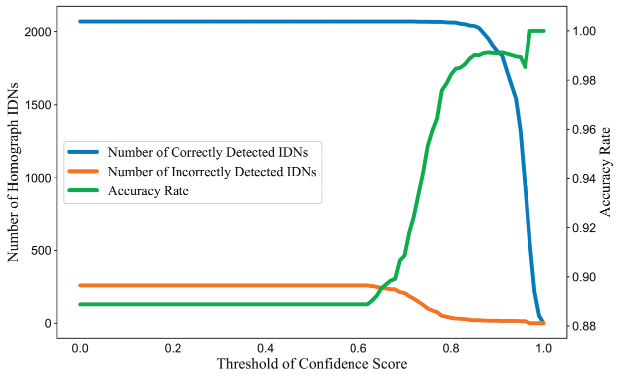


Fig. 6 Comparison of accuracy changing the threshold of confidence score.

Table 5 TLD of the detected homograph IDNs.

(a) Registered for domain protection

TLD	#IDNs
.com	108
.net	23
.org	16
.info	10
.biz	6
.fr	1
.pl	1
.services	1
Total	166

(b) Not registered for domain protection

TLD	#IDNs
.com	1,615
.net	95
.org	31
.tk	31
.de	24
.top	21
.ga	14
.xyz	13
.cf	12
.cc	10
.biz	10
.info	9
Others	137
Total	2,022

6. Case Study

In this section, we discuss the detected homograph IDNs based on the evaluation results presented in Section 5.

6.1 Homograph IDNs detected only by the conventional method

In the following section, we discuss homograph IDNs that were detected solely by the conventional method. **Figure 7** shows several examples of non-ASCII characters used in homograph IDNs that were detected by the conventional method but not by the proposed method. Our method was unable to detect these IDNs be-

Non-ASCII Characters	Visually Similar ASCII Characters	Number of Occurrences
ℸ	i	83
ℸ̈	i	30
ç	c	15
å	a	2
é	e	2
ë	e	2
ó	o	2

Fig. 7 Example of characters detected only by the conventional method.

cause the OCR results did not include the expected strings. In particular, several characters in this example are similar to linear letters, such as "`l`" and "`i`," resulting in low OCR accuracy. However, we believe that changing the mask type or algorithm for selecting the characters to be used in conversion mapping would increase the number of homographs that our method can detect. Given that both methods can be used together in practical applications, the existence of homographs that can only be detected by the conventional method should not cause any issues.

6.2 Homograph IDNs detected only by the proposed method

Figure 8 shows several examples of homograph IDNs that were detected only by the proposed method. All these IDNs were obtained from the Project Sonar dataset. Thus, our method was the first to detect these registered homograph IDNs. The first example uses "θ" as a replacement for "`e`," but the conversion table of the conventional method identifies "θ" to be similar to "`o`." Meanwhile, our method identified "θ" to be similar to both "`o`" and "`e`" using OCR, thereby allowing this IDN to be detected as a homograph. In the second example, the used non-ASCII character is a ligature of O and E. Regardless of the character's meaning, attackers focus on its visual appearance similar to that of "ce." The third to fifth examples use non-ASCII characters that are not very similar to the alphabetic characters they replaced. This is possibly because multiple homograph IDNs using visually similar characters are already registered. Hence, attackers have no choice but to use loosely related characters when registering new homograph IDNs. Nonetheless, our method can detect these domain names because it considers several reading results of the same character. Meanwhile, the sixth example uses phonetic symbols to construct a homograph, presumably because the attacker was restricted to selecting characters that had not been used yet.

Given these results, attackers analyze and use numerous non-ASCII characters when creating homograph IDNs, thereby rendering it extremely difficult to manually construct and update conversion mappings. However, the proposed method automatically constructs conversion mappings based on the input IDNs. Therefore, our method has the potential to be considerably effective.

7. Discussion

When evaluating the proposed method, we set the Tesseract OCR package to only output alphabetic characters and only considered IDNs with non-ASCII characters similar to alphabetic characters. In future research, we will also consider a method for detecting homograph domain names that use numbers or exploit the similarities between English letters.

Homograph Domains	Legitimate Domains
googlɵ.com	google.com
offiœ.com	office.com
sahşbinden.com	sahibinden.com
sahibindən.com	sahibinden.com
sʌmsung.com	samsung.com
gʘʘgle.com	google.com

Fig. 8 Example of homograph domains and their corresponding legitimate domains.

In our evaluation, we only used the top 1,000 results provided by Alexa Top Sites as legitimate domain names. However, in the case of typosquatting, even low-ranking domains can be abused [27]. Thus, it is highly possible that increasing the number of legitimate domain names in the input will produce more homograph IDNs to be detected. However, this would also increase the number of candidate strings that accidentally match with the legitimate domains, potentially leading to detection errors. We should reduce the number of strings that we extract from the OCR results for use in the conversion mapping while still identifying a method to achieve highly accurate conversion by employing this smaller pool of character mappings.

The proposed method leverages OCR for recognizing characters in IDNs to visually similar characters automatically. Therefore, the accuracy of this method depends on which OCR we use. Especially, when the accuracy of OCR increases, some characters might not be converted properly. Even in such cases, we can deal with this problem by changing the generation method of mask images.

8. Related Work

Homograph attacks were first mentioned in a study by Gabrilovich and Gontmakher [3]. This study discussed potential methods for future homograph attacks and their effects using a homograph attack against PairGain's domain name in April 2000 as an example. Owing to this attack, many readers believed the false information posted on the fake site, causing an increase in the stock price to 31% prior to a significant decrease and resulting in investors suffering critical investment losses. The researchers also suggested highlighting non-ASCII characters in the browser as a method of preventing homograph attacks.

Holgers et al. [28] also conducted an extensive research on homograph domain names. The researchers investigated the traffic passing over a university network, generated homograph domain names for the sites accessed by users, and investigated the name resolution results. They used static conversion mapping to generate homograph domain names by replacing alphanumeric characters with similar non-ASCII characters. However, we followed a different approach for identifying homograph IDNs associated with legitimate domain names and dynamically generated the conversion mapping using OCR.

To study homographs in the phishing context, Dhamija et al. [29] developed sites similar to phishing sites and investigated whether users can distinguish these sites from legitimate ones. The researchers found that the homograph domain names generated by replacing "`w`" with "`vv`" were the most challenging to identify, with 91% of users being unaware that the site was fake. Furthermore, when the researchers examined the factors that users inspect to determine the site validity, they found that 23% of users only considered the content of the websites and did not focus on the domain name.

Liu et al. [15] examined the registered IDNs based on several zone files managed by TLDs. The researchers devised a system for detecting homograph IDNs by evaluating the similarity between legitimate domain names and IDNs using a structural similarity (SSIM) index. Meanwhile, the proposed method is different in that we evaluate similarity by using OCR. Multiple OCR results contribute toward detecting various homograph IDNs.

After investigating typosquatting in more than seven months in 2013, Agten et al. [2] reported that several trademark owners are particular in obtaining possible typosquatting domain names related to their own domains for defense purposes. The researchers knew that more than 75% of possible typosquatting domains had already been acquired by well-known sites with relatively short names. Hence, they assumed that sites with long domain names would be the target in the future.

Szurdi et al. [27] demonstrated that typosquatting was also an issue for lower-rank domains in Alexa Top Sites and that approximately 20% of all "`.com`" domain name registrations can be attributed to typosquatting. They also discussed possible methods to prevent obtaining malicious domain names as well as techniques to detect typosquatting from the user perspective.

These methods used in the aforementioned studies are different from those used in our research. Our method can detect homograph IDNs corresponding to legitimate domains by employing IDNs and a list of legitimate domain names without the necessary advance preparation of conversion mapping.

9. Conclusion

In this study, we proposed a new method for detecting homograph IDNs using OCR. The conventional approach to detect such IDNs employs a predefined mapping table. However, this method has two significant limitations. First, it cannot detect homographs with characters that are not defined in the mapping. Second, mapping must be manually updated. To address these issues, we focused on the core feature of homograph IDNs (i.e., these IDNs are visually similar to legitimate domain names) and proposed a new method that dynamically creates a mapping that employs OCR to detect homographs. Then, we evaluated this method using 3.19 million real and 10,000 malicious IDNs and confirmed that our method can detect homograph IDNs that cannot be detected by the conventional methods. We also revealed that the countermeasures taken by legitimate domain holders are very inefficient; hence, attackers continue to generate homograph domain names. Therefore, the proposed method is effective in easily assessing whether the domain names accessed by users are homograph IDNs.

Acknowledgments A part of this work was supported by JSPS Grant-in-Aid for Scientific Research B, Grant Number JP16H02832.

References

[1] Sawabe, Y., Chiba, D., Akiyama, M. and Goto, S.: Detecting homograph IDNs using OCR, *Proc. Asia-Pacific Advanced Network* (*APAN*) (2018).

[2] Agten, P., Joosen, W., Piessens, F. and Nikiforakis, N.: Seven months' worth of mistakes: A longitudinal study of typosquatting abuse, *Proc. 22nd Network and Distributed System Security Symposium* (*NDSS 2015*), Internet Society (2015).

[3] Gabrilovich, E. and Gontmakher, A.: The homograph attack, *Comm. ACM*, Vol.45, No.2, p.128 (2002).

[4] Symatec: Bad guys using internationalized domain names (IDNs), available from ⟨https://www.symantec.com/connect/blogs/bad-guys-using-internationalized-domain-names-idns⟩.

[5] Ulikowski, M.: dnstwist, available from ⟨https://github.com/elceef/dnstwist/⟩.

[6] Unicode security mechanisms for utr #39 (2017), available from ⟨https://www.unicode.org/Public/security/10.0.0/confusables.txt⟩.

[7] ICANN: Internationalized domain names, available from ⟨https://www.icann.org/resources/pages/idn-2012-02-25-en⟩.

[8] Faltstrom, P., Hoffman, P. and Costello, A.: Internationalizing domain names in applications (IDNA), RFC 3490, RFC Editor (Mar. 2003).

[9] Hoffman, P. and Blanchet, M.: Nameprep: A stringprep profile for internationalized domain names (IDN), RFC 3491, RFC Editor (Mar. 2003).

[10] Hoffman, P. and Blanchet, M.: Preparation of internationalized strings ("stringprep"), RFC 3454, RFC Editor (Dec. 2002).

[11] Costello, A.: Punycode: A bootstring encoding of unicode for internationalized domain names in applications (IDNA), RFC 3492, RFC Editor (Mar. 2003).

[12] McElroy, T., Hannay, P. and Baatard, G.: The 2017 homograph browser attack mitigation survey (2017).

[13] Zheng, X.: Phishing with unicode domains (2017), available from ⟨https://www.xudongz.com/blog/2017/idn-phishing/⟩.

[14] Wordfence: Chrome and firefox phishing attack uses domains identical to known safe sites (2017), available from ⟨https://www.wordfence.com/blog/2017/04/chrome-firefox-unicode-phishing/⟩.

[15] Liu, B., Lu, C., Li, Z., et al.: A reexamination of internationalized domain names: The good, the bad and the ugly, *Proc. IEEE/IFIP Int. Conf. Dependable Systems and Networks* (*DSN*), pp.654–665 (2018).

[16] Public suffix list, available from ⟨https://publicsuffix.org/list/⟩.

[17] Chiba, D., Yagi, T., Akiyama, M., Shibahara, T., Mori, T. and Goto, S.: Domainprofiler: Toward accurate and early discovery of domain names abused in future, *International Journal of Information Security* (Dec. 2017).

[18] Rapid7: Project sonar forward dns (2017), available from ⟨https://opendata.rapid7.com/sonar.fdns_v2/⟩.

[19] hpHosts: Ad and tracking servers only, available from ⟨https://hosts-file.net/ad_servers.txt⟩.

[20] Dns-bh malware domain blockilist, available from ⟨http://www.malredomains.com/⟩.

[21] Phishtank, available from ⟨https://www.phishtank.com⟩.

[22] The spamhaus project Ltd.: the domain block list, available from ⟨https://www.spamhaus.org/dbl⟩.

[23] Uribl, available from ⟨https://uribl.com/about.html⟩.

[24] Alexa Internet: Alexa topsites, available from ⟨https://www.alexa.com/topsites⟩.

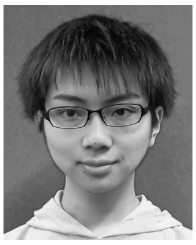
[25] Tesseract ocr, available from ⟨https://opensource.google.com/projects/tesseract/⟩.

[26] Smith, R.: An overview of the tesseract ocr engine, *Proc. Int. Conf. Document Analysis and Recognition* (*ICDAR*), Vol.2, pp.629–633 (2007).

[27] Szurdi, J., Kocso, B., Cseh, G., Spring, J., Felegyhazi, M. and Kanich, C.: The long "taile" of typosquatting domain names, *USENIX Security Symposium*, pp.191–206 (2014).

[28] Holgers, T., Watson, D.E. and Gribble, S.D.: Cutting through the confusion: A measurement study of homograph attacks, *USENIX Annual Technical Conference, General Track*, pp.261–266 (2006).

[29] Dhamija, R., Tygar, J.D. and Hearst, M.: Why phishing works, *Proc. SIGCHI Conference on Human Factors in Computing Systems*, pp.581–590, ACM (2006).



Yuta Sawabe received his B.S. and M.S. degrees in Computer Science and Engineering from Waseda University in 2017 and 2019. His research interest covers Cyber Security. He is now with NTT Security (Japan) KK, Tokyo, Japan.

Daiki Chiba is currently a researcher at NTT Secure Platform Laboratories, Tokyo, Japan. He received his B.E., M.E., and Ph.D. degrees in computer science from Waseda University in 2011, 2013, and 2017. Since joining Nippon Telegraph and Telephone Corporation (NTT) in 2013, he has been engaged in research on cyber security through data analysis. He won the Research Award from the IEICE Technical Committee on Information and Communication System Security in 2016 and the Best Paper Award from the IEICE Communications Society in 2017. He is a member of IEEE and IEICE.

Mitsuaki Akiyama received his M.E. and Ph.D. degrees in Information Science from Nara Institute of Science and Technology, Japan in 2007 and 2013. Since joining Nippon Telegraph and Telephone Corporation NTT in 2007, he has been engaged in the research and development of network security, especially honeypot and malware analysis. He is now with the Cyber Security Project of NTT Secure Platform Laboratories.

Shigeki Goto is a professor emeritus, Waseda University, Japan. He received his B.S. and M.S. in Mathematics from the University of Tokyo. He also earned his Ph.D in Information Engineering from the University of Tokyo. He has worked for NTT Laboratories from 1973 to 1996. He was a professor at Waseda University from 1996 to 2019. He is the president of JPNIC. He is a member of ACM and IEEE. He was inducted into the Internet Hall of Fame by Internet Society in 2017.