

Visual Spoofing Attacks and Defenses

An informational product and security white paper

August 2010
Casaba Security, LLC
www.casabasecurity.com

casaba

Table of Contents

1	OVERVIEW	3
2	USAGE SCENARIOS	3
2.1	ANTI-PHISHING PLATFORMS.....	4
2.2	FRAUDULENT VANITY URI'S.....	4
2.3	BYPASSING PROFANITY FILTERS.....	4
2.4	SPOOFING IN USER INTERFACE AND DIALOGS.....	4
2.5	MALVERTISEMENTS IN ADVERTISING NETWORKS.....	5
2.6	FORGING INTERNATIONALIZED EMAIL.....	5
3	PRIOR RESEARCH	5
4	THE WEB BROWSER SCENARIO: VISUALLY CONFUSABLE DOMAIN NAMES	6
4.1	SINGLE, MIXED, AND WHOLE-SCRIPT SPOOFING	6
4.2	SUB-DOMAIN LABEL SPOOFING.....	7
4.3	COMBINING MARKS SPOOFING.....	8
4.3.1	Combining Mark Tricks	8
4.4	BIDI AND THE INVISIBLES	9
4.5	IDNA2003 AND IDNA2008.....	10
5	A SOLUTION: UCAPI FOR CONFUSABLE AND VISUAL SIMILARITY DETECTION	10
5.1	ADD-ON BENEFITS.....	11
5.2	HOW IT WORKS.....	11
6	REFERENCES	12
7	CONTACT	12

1 Overview

While Unicode has provided a useful framework for storing, transmitting, and presenting information in many of our world's native languages, it has also better enabled attack vectors for phishing and word filters. Commonly referred to as 'visual spoofing' these attack vectors use letters and numbers from various languages that are visually identical to letters in another language.

To a human reader, some of the following letters are indistinguishable from one another while others closely resemble one another:

AAAAΔAΛ

To a computer system however, each of these letters has very different meaning. The underlying bits that represent each letter are different from one to the next. How then, could a software vendor possibly implement a solution which guarantees expected visual appearances?

A core library solution is one apparent tool needed to give software vendors options for the scenarios which exploit visual spoofing. Casaba Security, LLC (www.casabasecurity.com) has developed just such a cross-platform library in C and C++. This library thwarts fraudulent visual spoofing attacks by identifying visually confusable characters and comparing similar strings across many languages.

2 Usage Scenarios

Since the advent of Internationalized Domain Names (IDN), visually confusable characters have become a significant threat in the area of phishing and other fraudulent attacks against Internet users. Web browsers are often the vehicle for exploiting these attacks, known as *visual spoofing attacks*. While IDN seems the most obvious place for visual fraud, these attacks can be extended much more broadly to target content, User-Interfaces, word filters (e.g. profanity), usernames in LDAP/directory servers, and anywhere else visual appearance is pivotal to user decision-making and security.

In each of these cases, the UCAPI library can be used to detect attacks based on:

- **Analyzing a single string for confusability**
Sometimes a fast string scan may be all that's required to detect the presence of confusable characters, invisibles, multiple combining marks, or other potential attacks.
- **Comparing two strings for visual similarity**
In other cases, it may be more appropriate to compare two strings. In the case of a vanity URL for example, a Web-platform may want to determine if new URL registrations are visually similar to existing ones before deciding to allow them.

Consider some of the following scenarios, and see later in this document for an overview of [how UCAPI works](#).

2.1 Anti-Phishing Platforms

Phishing attacks continue to evolve with email spam campaigns that look more and more like they're coming from authentic sources. As IDN's continue to become more mainstream phishers will have renewed ammunition at their disposal to craft fraudulent messages and domain names that look visually identical to their legitimate counterparts. Anti-phishing platforms could implement confusability detection now to start collecting data and gaining insight into these attack vectors - before a widespread attack occurs.

2.2 Fraudulent Vanity URI's

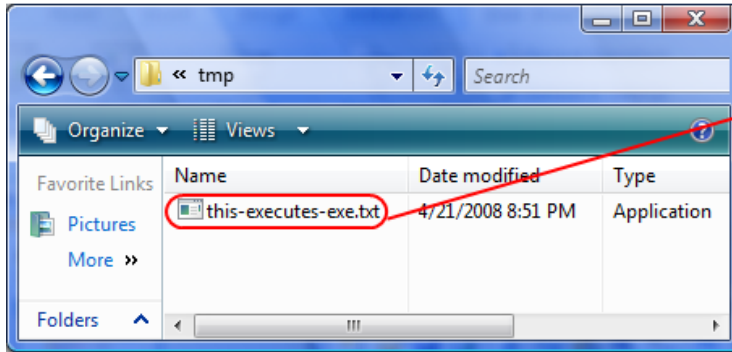
A social networking service wants to allow vanity URI's to be registered using international characters such as [www.foo.bar/дежз](#) but cannot for fear of all the ways that vanity URI's could be subject to visual fraud and confusion. Because Unicode characters are well supported in the path portion of a browser's URI display, a well-crafted vanity URI could easily fool victims and be the landing page for a phishing attack.

2.3 Bypassing Profanity Filters

Another scenario is an email or forum system that needs to prevent violent and profane words from being used. There are trivial ways to bypass such filters, including using spacing and punctuation between letters in a word (e.g. c_r_a_p), slight misspellings which give the same effect (e.g. crrap), and then there's using confusable characters which have no visual side-affect (e.g. crap) written entirely in another script (or mix thereof).

2.4 Spoofing in User Interface and Dialogs

Many security decisions are given to end users in the form of dialog boxes. For example, when a user downloads a file through a Web browser, they're asked to confirm their decision. When a user tries to launch an untrusted application they may also be presented with a dialog box asking for confirmation. A social networking site may ask its users for confirmation before redirecting them to an off-site URL. A clever attack may use special BIDI or other characters that reverse the direction of text to fool end users.



this-executes-[U+202E]txt.exe

2.5 Malvertisements in Advertising Networks

Consider an advertising network that needs to mitigate malicious ads by protecting brand name trademarks from being registered by anyone other than their owner. An attacker could place an ad that bypasses trademark filters using confusable characters in order to fool users into visiting a phishing site. For example “Download Microsoft Service Pack 1 for Windows 7 here” where the trademarked name ‘Microsoft’ was crafted using non-English script.

2.6 Forging Internationalized Email

Email addresses have long been confined to ASCII but there will most certainly be a time when they’re opened up to UTF-8 and international characters. In preparing for that transition, email client designers will need to anticipate and handle the case of visually identical email addresses. If those cases are not handled, then end users could easily be fooled. Digital certificates would provide a good mechanism for proving authenticity of a message; however such certificates also support Unicode and are vulnerable to the exact same attacks.

3 Prior Research

One of the most well-known attacks to exploit visual spoofing was the Paypal.com IDN spoof of 2005. Setup to demonstrate the power of these attack vectors, [Eric Johanson](#) and The Schmo Group successfully used a [www.paypal.com](#) lookalike domain name to fool visitors into providing personal information. The advisory references original research from 2002 by Evgeniy Gabrilovich and Alex Gontmakher at the Israel Institute of Technology. Their [original paper](#) described an attack using Microsoft.com as an example.

Viktor Krammer, author of the [Quero Toolbar](#) for Internet Explorer, also presented additional research on these attack vectors and detection mechanisms in his [2006 presentation](#). Additionally, the [Unicode Consortium](#) has been active at raising awareness of these issues in its security papers, and in providing recommended solutions.

4 The Web Browser Scenario: Visually Confusable Domain Names

To run with an example, IDN's make a good case study. Visual spoofing attacks leveraging IDNs are poised to become a significant problem as IDNs gain in popularity. Although data published by the Anti-Phishing Working Group (APWG) has not shown IDNs as a significant vector for attacks in the past, Casaba Security believes that the surface for such attacks is indeed significant, and will only continue to widen as IDN ccTLDs and gTLDs emerge through ICANN. We believe that a sharp increase in malicious IDN activity can be expected in the coming years.

Of particular note (giving credence to this assertion), is that registering domain names which are visually confusable is trivial. A well-placed attack could leverage a domain name that appears visually identical to a well-known brand, bank, or other trademark like the following:

1. www.microsoft.com
2. www.bankofamerica.com
3. www.google.com

Domain names can be visually spoofed through a number of methods which will be summarized here.

4.1 Single, Mixed, and Whole-script Spoofing

IDNs enable domain names to contain scripts from many of the world's languages. In general, there are three ways to categorize a domain name spoof that leverages Unicode characters:

1. Single-script Spoof

Letters from the same alphabet, or script, are used to give the same visual appearance. This definition should be extended to say that these occur when letters from either the same script, inherited script, or common script, are used together (as defined by the Unicode standard). For example, the following domain name spoof uses all Latin letters with the extended Latin small letter Alpha 'ɑ':

www.apple.com

2. Mixed-script Spoof

Letters from two or more different alphabets, or scripts, are used to give the same visual appearance as letters from other scripts. For example, the following domain name mixes Latin with the Greek lunate sigma symbol 'ς':

<http://www.facebook.com>

3. Whole-script Spoof

Letters are made up entirely from one script that is visually confusable with letters of another script. For example, the following domain name is made entirely of Cyrillic

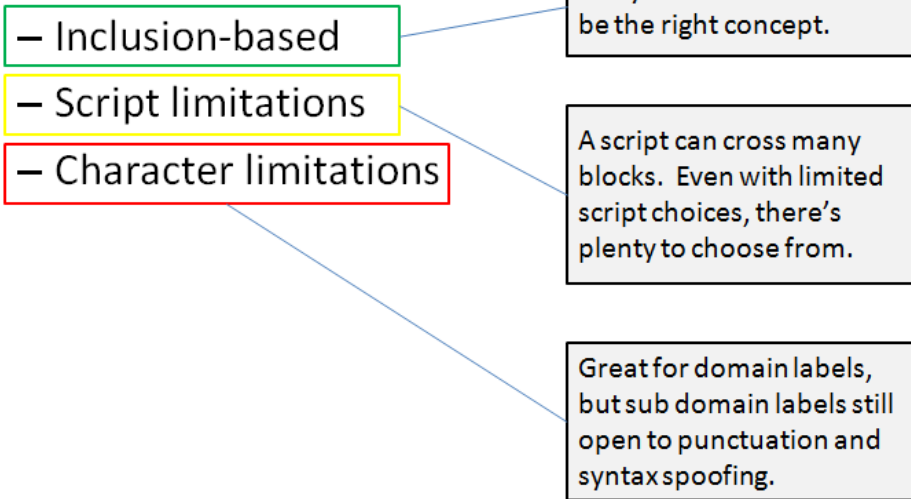
script confusable with the same word 'abc' entirely in Latin script:

www.abc.com

The concept of the visual spoofing threat has been known for some time, and certain mitigations have been attempted by the standards bodies and registries.

For example, certain TLD's, such as .org, are subject to character restrictions as recommended by ICANN and implemented by the registry. These restrictions, summarized in the following illustration, are supposed to make domain names less vulnerable to visual spoofing attacks.

ICANN guidelines v2.0



While the concept has been well-known, the specific vectors and instances of attack might not be.

4.2 Sub-domain Label Spoofing

Even with the above mentioned guidance, sub-domain labels can still be exploited, and cannot be mitigated by the registries. For example, consider the following confusable domain name:

<http://www.google.com/path/file.nottrusted.org>

Katakana No
U+FF89

A working link is also provided for testing in this document:

<http://www.yahoo.com/path/to/file.php/phppage.nottrusted.org/>

In this case, the true domain name is *nottrusted.org*. However, the domain could visually appear as www.yahoo.com to many Internet users who might be confused by the KATAKANA LETTER NO character as a path separator. This demonstrates just one instance of this type of attack.

There are many other tricks that can be used to spoof syntax-punctuation such as dots (full stop), slashes (solidus), some ligatures, and combining marks.

4.3 Combining Marks Spoofing

Aside from purely confusable letters across scripts, there are yet many other ways to create visually confusable strings such as domain names.

4.3.1 Combining Mark Tricks

The visual attacks leveraging combining marks range from simple to complex and in many cases are dependent on the font for the final display. The following attack vectors exist:

1. Multiple Identical Combining Marks

Multiple identical combining marks can be spoofed in linear. For example, with a U+0065 LATIN SMALL LETTER E and a U+0301 COMBINING ACUTE ACCENT, the mark will visually sit on top of the character preceding it to give the appearance of the following (the Unicode code points are shown following the character):

é \u0065\u0301

This would be a normal use case of a combining mark. In the following example, we have a visual attack using two *identical* combining marks:

é \u0065\u0301\u0301

With the addition of a second combining mark, the byte structure and value of the string changes, but visually it looks the same.

***Note this is shown using the Arial Unicode MS font, and will appear different in different fonts.**

2. Multiple Different Combining Marks (and reordering tricks)

This class of spoofing is much harder to detect. Multiple, but different combining marks can be used to stack in layers, and give the visual appearance of a single combining mark. Consider the following combining marks:

- U+0335 COMBINING SHORT STROKE OVERLAY

- U+0336 COMBINING LONG STROKE OVERLAY

Each has a very different visual appearance – one is long and one is short. However, when combined, it seems impossible to visually see that there’s a short stroke hidden under the long stroke.

Consider the normal use case of a LATIN LETTER SMALL O with a combining long stroke:

⓪ \u006F\u0336

This is visually indistinguishable from the following spoofs, where an additional short stroke has been added. In fact, each of the cases below in which only the ordering of the code point sequence has changed:

⓪ \u006F\u0335\u0336

⓪ \u006F\u0336\u0335

3. Inadequate Rendering of Combining Mark

Another issue is combining marks that don’t display properly in certain fonts. For example, in the word “google” below, the combining mark between the ‘g’ and the ‘l’ is hidden from view in the Arial Unicode MS font. Several other fonts exhibit this behavior.

google

Shown in the Arial font, the same string clearly depicts the combining mark:

google

Combinations of the above variations make for many possibilities in specific attacks.

4.4 Bidi and The Invisibles

Bi-directional text can lead to spoofing problems in strings. In particular, the RIGHT-TO-LEFT and LEFT-TO-RIGHT OVERRIDE characters can force the direction of a string to change. So, by inserting a RIGHT-TO-LEFT OVERRIDE character between the ‘o’ and the ‘e’, the word “gooelg” can be made to look like “google”.

Invisibles represent another type of character which have no visual glyph, and do not take up white space. The ZERO WIDTH JOINER is one example of this type of character. By inserting an

invisible character into a string, the meaning of the string changes while the visual appearance does not.

In strings used in UI Dialog boxes and other places of an application, these characters can represent a significant threat. They are less of a threat in domain names as the IDNA2003 standard forbids them. However, IDNA2008 aims to relax some of these rules and allow certain invisible characters.

4.5 IDNA2003 and IDNA2008

Web browsers today currently implement the IDNA2003 standard for processing IDNs. This standard allows for symbols and other marks that resemble punctuation, enabling attacks in the subdomain field.

The standard also has other issues, which have theoretically been addressed, in the new IDNA2008 standard. IDNA2008 however, still in Fast Track, has its own issues:

- Incompatibility with IDNA2003
- Significant complexity
 - Allows for context-based use of ZERO WIDTH JOINER and NON-JOINER (invisible characters)
- Deviations
 - **ß** LATIN SMALL LETTER SHARP S resolves to different IP addresses under each standard

The expected implementation of IDNA2008 is likely years away, and we'll continue to live with the IDNA2003 standard until then. Once implemented, however, these new issues will emerge, and the same visual spoofing threats will still exist.

5 A Solution: UCAPI for Confusable and Visual Similarity Detection

Casaba Security, LLC has developed a solution which *transcends the discussed issues* with its detection of confusable characters and visually similar strings. UCAPI has been built for *portability* and *speed*. With a goal of cross-platform use, UCAPI was developed in C/C++ and tested on Mac OSX as well as Windows. In terms of speed, UCAPI was developed with the goal that its use should have no impact on the hosting application's performance. With this in mind, single-string analyses have been clocked at *sub-nanosecond speeds*.

With such a small footprint, UCAPI could fit right into a product's phishing and malware protection features to both:

1. Detect confusable characters in usernames, domain names, User Interface dialogs, vanity URLs, and other text-based content and;
2. Compare two strings for visual similarity.

Consider a case where a Web browser may want to protect its 10,000 most popular domain names specifically, by identifying attacks that are visually similar to them. The following scenario depicts how this could look in practice:

1. A user has enabled phishing and malware protection in the browser.
2. A user navigates to a domain name which is a visual spoof of “www.google.de” which was not listed in the database of known phishing sites.
3. In the case of “google” above, the letter ‘g’ is a visually confusable character.¹
4. With confusable-detection built into anti-phishing and anti-malware platform, the user could be immediately notified of the potential threat.

Additional benefits could be extended to components that require visual assurances for security decisions. For some examples, consider the following places where visual integrity plays an important factor:

- A User-Interface dialog box prompting the user to download a file.
- An SSL certificate name.
- A vanity URL on a popular social networking site.

The UCAPI solution for Confusable and Visual Similarity Detection comprises many approaches that can be configured together or individually to provide the desired level of accuracy and leniency.

5.1 Add-on Benefits

Additional benefits exist from implementing UCAPI’s confusable string protections:

1. Data collection providing more insight into this attack vector
2. Confidence to display pure IDN for .COM and other ‘dangerous’ TLD’s
3. Protection in other software components that need visual integrity guarantees

As the Unicode and IDNA standards evolve, implementing these features on back-end anti-phishing platforms would provide immediate benefit to end users.

5.2 How it Works

Casaba’s Confusable and Visual Similarity Detection component currently provides mitigation and a roadmap for coverage for many of the threats described in this document, in addition to other visual spoofing attack vectors not mentioned herein. Among other techniques, this solution implements the specification from UTR-39 the Unicode Consortium’s recommended approach to detecting string confusability.

¹ Note that the techniques for creating visually similar words extend far beyond simply mixing script from various languages.

These approaches leverage an evolving database of confusable characters to scan a given string and detect when confusability exists. Of course, at some level almost everything would seem confusable with something. For this reason, the UCAPI interface can be configured to reduce the set of characters under consideration based on script and language settings.

UCAPI can also compare two strings for visual similarity. While this process starts by raising each string to a common form for comparison, it's not about lower-casing. Beginning by decomposing the string with a common normalization form, UCAPI then applies character mapping and performs a character-by-character and overall string comparison using a set of pre-configured rules.

6 References

The following references have been useful in exploring these areas and creating this paper:

1. Viktor Krammer: [Phishing Defense against IDN Address Spoofing Attacks](#)
In Proceedings of the 4th Annual Privacy Security Trust Conference 2006 (PST 2006), pp. 275-284, ACM, October 2006
2. Unicode Consortium
[UTR# 36: Unicode Security Considerations](#)
[UTR# 39: Unicode Security Mechanisms](#)
[UTR# 46: IDNA Compatibility Processing](#)
3. Eric Johanson and The Schmoo Group
<http://www.shmoo.com/idn/>
4. Evgeniy Gabrilovich and Alex Gontmakher of the Israel Institute of Technology
<http://www.cs.technion.ac.il/~gabr/papers/homograph.html>

7 Contact

Please contact the following person(s) at Casaba Security for further discussion:

Chris Weber

Phone: (949) 637-4155

LinkedIn: <http://www.linkedin.com/in/chrisweber>

Email: chris@casabasecurity.com