Deep Confusables

Improving Unicode Encoding Attacks with Deep Learning

Miguel Hernández (@MiguelHzBz)
José Ignacio Escribano (@jiep)
Dr. Alfonso Muñoz (@mindcrypt)

May 2019

HITBSecConf

Who We Are

- **Miguel Hernández** (@MiguelHzBz)
- Security Researcher at BBVA Next Technologies
- Speaker at RootedCon, NoConName, Cybercamp...
- `miguel.hernandez2.next@bbva.com`

- **José Ignacio Escribano** (@jiep)
- ML & Security Researcher at BBVA Next Technologies
- `joseignacio.escribano.pablos.next@bbva.com`

Outline

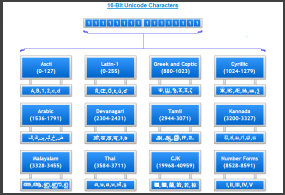
1. Unicode 101
   - Basics
   - Unicode and confusables
   - Punycode
2. Related work
   - Related researches
   - Open Source Tools
3. Deep Learning and Security
   - Some examples
4. Deep Confusables
5. Unicode attacks in real world
6. Countermeasures
7. Conclusions

Table of contents

1. Unicode 101
   - Basics
   - Unicode and confusables
   - Punycode
2. Related work
   - Related researches
   - Open Source Tools
3. Deep Learning and Security
   - Some examples
4. Deep Confusables
5. Unicode attacks in real world
6. Countermeasures
7. Conclusions

Unicode

- An international encoding standard for use with different languages and scripts, by which each letter, digit, or symbol is assigned a unique numeric value that applies across different platforms and programs.
- Unicode[1] 12.0 adds 554 characters, for a total of 137,928 characters.



[1] https://www.unicode.org

Unicode and confusables

New phishing scam masquerades as Apple support call

By Roger Fingas
Friday, January 04, 2019, 02:06 pm PT (05:06 pm ET)

The latest scam targeting Apple device users is particularly insidious, appearing to come as a call from the company's real phone support number, according to a well-known security researcher.

PlayStation 4 reportedly crashing due to malicious message

Here's how to protect your PS4 against the bug

By Chris Welch | @chriswelch | Oct 13, 2018, 8:05pm EDT

What is a confusable?

http://example.org

http://example.org

HITB

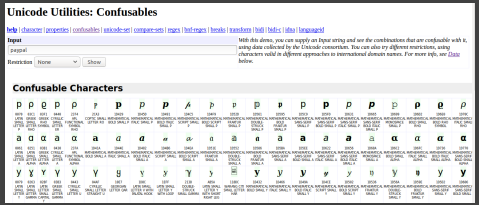
What is a confusable?

http://example.org

http://www.xn–xample-2of.org/

Confusables provided by Unicode Consortium

With this demo[1], you can supply an Input string and see the combinations that are confusable with it, using data collected by the Unicode consortium. You can also try different restrictions, using characters valid in different approaches to international domain names.



[1] https://unicode.org/cldr/utility/confusables.jsp

Punycode

Unicode Security Mechanisms[1]

[1] http://www.unicode.org/reports/tr39/tr39-19.html

Punycode (RFC3492) is a representation of Unicode with the limited ASCII character subset used for IDNA (internationalized domain names). Unicode characters are transcoded to a subset of ASCII (consisting of letters, digits, and hyphen, which is called the Letter-Digit-Hyphen, LDH subset) favored by DNS

Confusables provided by Unicode Consortium - FAQs

Q: How serious is the problem of spoofing with Unicode characters?

A: It is important to recognize that the use of visually confusable characters in spoofing is often overstated. Confusable characters account for a small proportion of phishing problems: most instances of phishing involve social engineering or simple misleading domain names such as "secure-wellsfargo.com". For more information, see http://www.bortzmeyer.org/idn-et-phishing.html. (It is in French, but you can use Google translate or other services to get the gist of the document if you don't read French.)

HITB

Table of contents

1. Unicode 101
   - Basics
   - Unicode and confusables
   - Punycode
2. Related work
   - Related researches
   - Open Source Tools
3. Deep Learning and Security
   - Some examples
4. Deep Confusables
5. Unicode attacks in real world
6. Countermeasures
7. Conclusions

Related work

Irongeek (2017)
http://www.irongeek.com/i.php?page=security/out-of-character-use-of-punycode-and-homoglyph-attacks-to-obfuscate-urls-for-phishing

DEF CON 26 (2018) - The Tarquin - Weaponizing Unicode Homographs Beyond IDNs
https://www.defcon.org/html/defcon-26/dc-26-speakers.html#Tarquin

Tools – EvilURL

EvilURL[1]. Generate unicode evil domains for IDN Homograph Attack and detect them.

```
if "A" in url.upper():
    makeEvil('a', names[0], unicodes[0], url.replace('a', '\u0430'), end)
    urlMore = urlMore.replace('a', '\u0430')
    urlChars += 'a, '
    urlNms += names[0] + ', '
    urlUn += unicodes[0] + ", "

if "C" in url.upper():
    makeEvil('c', names[1], unicodes[1], url.replace('c', '\u03F2'), end)
    urlMore = urlMore.replace('c','\u03F2')
    urlChars += 'c, '
    urlNms += names[1] + ', '
    urlUn += unicodes[1] + ", "

if "E" in url.upper():
    makeEvil('e', names[2], unicodes[2], url.replace('e', '\u0435'), end)
    urlMore = urlMore.replace('e', '\u0435')
    urlChars += 'e, '
    urlNms += names[2] + ', '
    urlUn += unicodes[2] + ", "

if "O" in url.upper():
    makeEvil('o', names[3], unicodes[3], url.replace('o', '\u043E'), end)
    urlMore = urlMore.replace('o', '\u043E')
    urlChars += 'o, '
    urlNms += names[3] + ', '
    urlUn += unicodes[3] + ", "

if "P" in url.upper():
    makeEvil('p', names[4], unicodes[4], url.replace('p', '\u0440'), end)
```

[1] https://github.com/UndeadSec/EvilURL

Tools – Squatm3

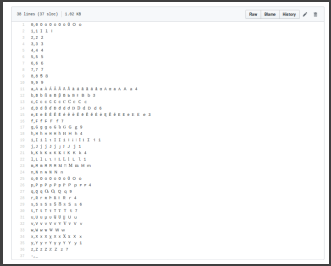
Squatm3[1] is a Python tool designed to enumerate available domains generated modifying the original domain name through different techniques:

- Substitution attack.
- Flipping attack.
- **Homoglyph attack** fast (execute a fast homoglyph attack, mutating only one letter at the time).
- **Homoglyph attack** complete (generates all the possible combinations).

[1] https://github.com/david3107/squatm3

Squatm3 – Confusables provided

Squatm3[1]



[1] https://github.com/david3107/squatm3/blob/master/db/homoglyph

Tools – Samesame

Samesame[1] is a lightweight utility for replacing ASCII characters with homograph (look-alike) characters.

```
use self::rand::Rng;

use std::string::String;
use std::collections::HashMap;

pub fn map(input: &str) -> String {
    let mut confusables = HashMap::new();

    confusables.insert('A', vec!["\u{0410}","\u{1D00}","\u{1EA0}",
        "\u{1EA2}","\u{1EA4}","\u{1EA6}","\u{1EA8}","\u{1EAA}","\u{1EAC}","\u{1EAE}","\u{1EB0}",
        "\u{1EB2}","\u{1EB4}","\u{1EB6}","\u{0391}","\u{1D0A}","\u{1D34}","\u{1D00}",
        "\u{1D400}","\u{1D434}","\u{1D468}","\u{1D49C}","\u{1D4D0}","\u{1D504}","\u{1D538}","\u{1D670}",
        "\u{1D6A8}","\u{1D6E2}","\u{1D71C}","\u{1D56C}","\u{1D5A0}","\u{1D5D4}",
        "\u{1D608}","\u{1D642}","\u{1D67C}","\u{1D4B6}"]);

    confusables.insert('B', vec!["\u{0532}","\u{212C}","\u{0042}","\u{1D00}","\u{1D0B}","\u{1D34}","\u{1D001}",
        "\u{0392}","\u{1D6A9}","\u{1D6E3}","\u{1D71D}","\u{1D57D}","\u{1D5B1}","\u{1D5E5}","\u{1D619}",
        "\u{1D653}","\u{1D68D}","\u{1D4B7}"]);

    confusables.insert('C', vec!["\u{0297}","\u{1D04}","\u{1D9C}","\u{2102}","\u{2103}","\u{2105}","\u{2109}",
        "\u{212D}","\u{1D00C}","\u{1D436}","\u{1D470}","\u{1D4AA}","\u{1D4D2}","\u{1D50C}","\u{1D540}",
        "\u{1D57A}","\u{1D5B4}","\u{1D5EE}","\u{1D628}","\u{1D662}","\u{1D69C}","\u{1D6D0}","\u{1D70A}",
        "\u{1D744}","\u{1D4B8}"]);

    confusables.insert('D', vec!["\u{2145}","\u{1D4AA}","\u{1D007}","\u{1D00D}","\u{1D4D3}","\u{1D50D}",
        "\u{1D541}","\u{1D57B}","\u{1D5B5}","\u{1D5E9}","\u{1D61D}","\u{1D651}","\u{1D685}","\u{1D4B9}"]);

    confusables.insert('E', vec!["\u{0415}","\u{1D07}","\u{1D0E}","\u{1D34A}","\u{1D00E}","\u{1D4A4}","\u{1D4D4}",
        "\u{1D518}","\u{1D552}","\u{1D58C}","\u{1D5C6}","\u{1D600}","\u{1D63A}","\u{1D674}","\u{1D6AE}",
        "\u{1D6E8}","\u{1D722}","\u{1D5C0}","\u{1D4BA}"]);

    confusables.insert('F', vec!["\u{0493}","\u{1D0F}","\u{1D00F}","\u{1D4A5}","\u{1D4D5}","\u{1D519}",
        "\u{1D553}","\u{1D58D}","\u{1D5C7}","\u{1D601}","\u{1D63B}","\u{1D675}","\u{1D6AF}","\u{1D5C1}",
        "\u{1D4BB}"]);

    confusables.insert('G', vec!["\u{050C}","\u{0262}","\u{1D33}","\u{1D010}","\u{1D4A6}","\u{1D4D6}",
        "\u{1D51A}","\u{1D554}","\u{1D58E}","\u{1D5C8}","\u{1D602}","\u{1D63C}","\u{1D676}","\u{1D6B0}",
        "\u{1D5C2}","\u{1D4BC}"]);

    confusables.insert('H', vec!["\u{0397}","\u{1D34F}","\u{1D013}","\u{1D4A7}","\u{1D4D7}","\u{1D51B}",
        "\u{1D555}","\u{1D58F}","\u{1D5C9}","\u{1D603}","\u{1D63D}","\u{1D677}","\u{1D6B1}","\u{1D5C3}",
        "\u{1D4BD}"]);

    confusables.insert('I', vec!["\u{0406}","\u{1D0A}","\u{1D35}","\u{1D02C}","\u{1D4A8}","\u{1D4D8}",
        "\u{1D51C}","\u{1D556}","\u{1D590}","\u{1D5CA}","\u{1D604}","\u{1D63E}","\u{1D678}","\u{1D6B2}",
        "\u{1D5C4}","\u{1D4BE}"]);

    confusables.insert('J', vec!["\u{0408}","\u{1D0B}","\u{1D36}","\u{1D02D}","\u{1D4A9}","\u{1D4D9}",
        "\u{1D51D}","\u{1D557}","\u{1D591}","\u{1D5CB}","\u{1D605}","\u{1D63F}","\u{1D679}","\u{1D6B3}",
        "\u{1D5C5}","\u{1D4BF}"]);

    confusables.insert('K', vec!["\u{212A}","\u{0199}","\u{1D04B}","\u{1D037}","\u{1D4AA}","\u{1D4DA}",
        "\u{1D51E}","\u{1D558}","\u{1D592}","\u{1D5CC}","\u{1D606}","\u{1D640}","\u{1D67A}","\u{1D6B4}",
```

[1] https://github.com/TheTarquin/samesame

Table of contents

1. Unicode 101
   - Basics
   - Unicode and confusables
   - Punycode
2. Related work
   - Related researches
   - Open Source Tools
3. Deep Learning and Security
   - Some examples
4. Deep Confusables
5. Unicode attacks in real world
6. Countermeasures
7. Conclusions

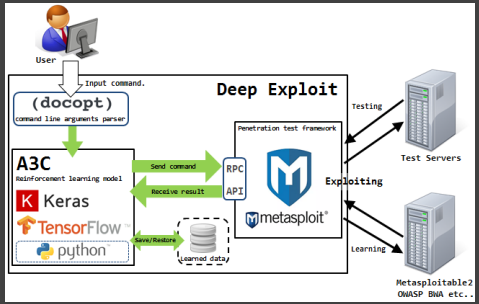
Deep Learning and Security

Why Deep Learning is used in this research?

- Machine Learning and Deep Learning can be used to create offensive and defensive tools.
- It improves human performance on some tasks.
- An increasing amount of hacking tools are appearing in the last years.

DeepExploit

DeepExploit[1]. Fully automatic penetration test tool using Machine Learning.



[1] https://git.io/fj3Yv

PassGAN

PassGAN[1]. This repository contains code for the PassGAN: A Deep Learning Approach for Password Guessing paper.

TABLE IV: Sample of passwords generated by the GAN that did not match the testing set.

love42743	ilovey2b93	paolo9630	italyit
sadgross	usa2598	s13trumpy	trumpart3
ttybaby5	dark1106	vamperiosa	~dracula
saddracula	luvengland	albania.	bananabake
paleyoung	@crepess	emily1015	enemy20
goku476	coolarse18	iscoolin	serious003
nyc1234	thepotus12	greatrun	babybad528
santazone	apple8487	1loveyoung	bitchin706
toshibaod	tweet1997b	103tears	1holys01

[1] https://github.com/brannondorsey/PassGAN

Table of contents

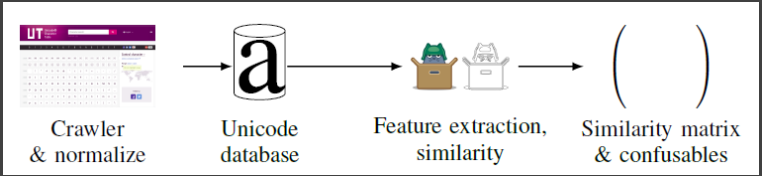
1. Unicode 101
   - Basics
   - Unicode and confusables
   - Punycode
2. Related work
   - Related researches
   - Open Source Tools
3. Deep Learning and Security
   - Some examples
4. Deep Confusables
5. Unicode attacks in real world
6. Countermeasures
7. Conclusions

What is Deep Confusables?

- System to obtain new Unicode confusables using deep learning from Latin characters.

- It works in an automated way.
- Composed of three components:
  1. Unicode image database.
  2. Feature extractor and similarity comparator.
  3. Command Line Interface (CLI).

How it works



- Crawler and normalization.
- Feature extraction and similarity comparator.
- Get similarity matrix and confusables.
- Confusables are used by CLI (based on threshold).

Unicode image database

- Images extracted from `https://unicode-table.com`.
- Images have been normalized to 34x34 pixels.
- 38,800 images from 266 Unicode blocks.

`https://github.com/next-security-lab/unicode-images-database/releases`

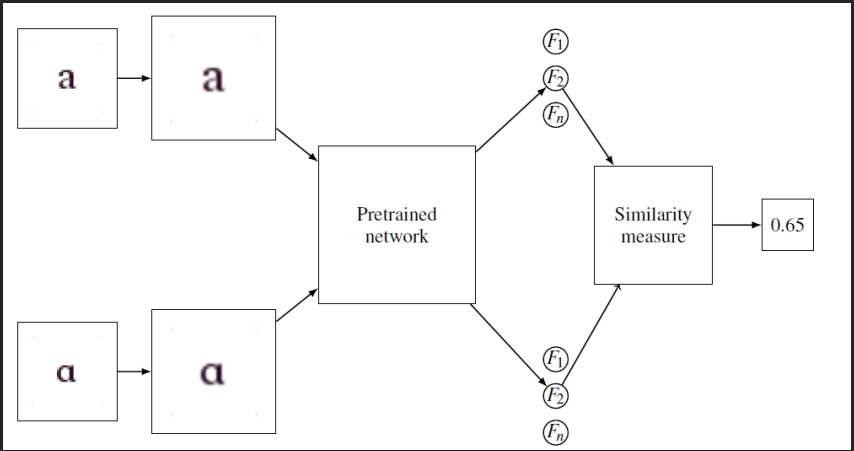
HITB

Feature extraction and similarity comparator

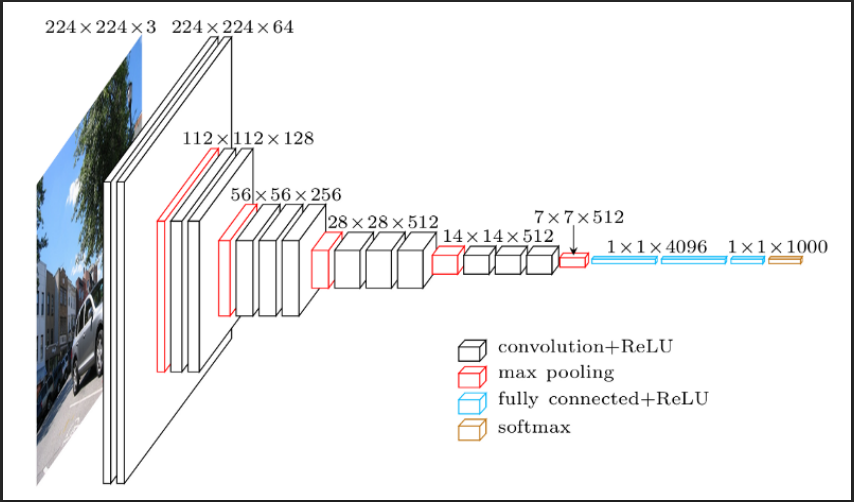
1. Obtain similarity matrix for each Latin character.
   - For each Latin character and non Latin characters:
     - Extract features using deep learning with a pretrained model (e.g. VGG 16).
     - Compare features using similarity function (e.g. cosine similarity).
2. Get confusables.
   - Fix a threshold θ between $0 \leq \theta \leq 1$.
   - Obtain Unicode characters whose similarity is greater than θ.

https://github.com/next-security-lab/deep-confusables-similarity/releases

Feature extraction and similarity comparator – Scheme



VGG16


224 × 224 × 3
224 × 224 × 64
112 × 112 × 128
56 × 56 × 256
28 × 28 × 512
14 × 14 × 512
7 × 7 × 512
1 × 1 × 4096
1 × 1 × 1000
convolution+ReLU
max pooling
fully connected+ReLU
softmax

HITB

Feature extraction and similarity comparator

Cosine similarity

$$\cos(x, y) = \frac{\langle x, y \rangle}{\|x\| \cdot \|y\|} \tag{1}$$

HITB

Command Line Interface

- Creates new domains given a threshold using confusables.
- Only Latin characters supported.
- Some features.
  - Check if domains are up.
  - Check Whois.
  - Check domain against VirusTotal (API key needed).

https://github.com/next-security-lab/deep-confusables-cli

Command Line Interface – Some use cases

1. Offensive tool
   - Red team.
   - APT.
   - Impersonate domains.
   - Phishing campaigns.
2. Defensive tool
   - Blue team.
   - Detect similar domains to ours.
   - Register similar domains.

Command Line Interface – Demo

DeepConfusables

```
user@host$ d
```

Deep Confusables dictionary

- We have combined static dictionaries from other tools with our confusables with threshold 75%.
- It can be used in some use cases.

https://github.com/next-security-lab/deep-confusables-cli/blob/master/deep_confusables_lite/confusables.txt

HITB

UriDeep

- It is like DeepConfusables CLI, but faster.
- Based on Deep Confusables dictionary.
- It can reduce false positive rate.
- It includes additional functionality
  - Substitution attack.
  - Flipping attack.

https://github.com/mindcrypt/uriDeep
https://github.com/mindcrypt/uriDeep/blob/master/data/deepDiccConfusables.txt

UriDeep – Demo

UriDeep

user@host

HITB

Some results

- Top 10,000 domains from Alexa analyzed.
- Domains generated with Deep Confusables dictionary.
- 54 confusables per character on average.
- 27,876 domains are up.

HITB

Some examples

amazón.es, góogle.es skypê.net, skýpe.net, skÿpe.net, skypè.net, skypé.net, fàcebook.net, fâcebook.net, facêbook.net, facëbook.net, mınecraft.net, twīt-ter.com, t-mobìle.com, alıexpress.com, applē.com, îkea.com, braźzers.com, īnstagram.com, netflìx.com, facebook.com, the uardian.com, ebáy.com, americanexpress.com, adïdas.com, sėx.com, whatsàpp.com, àirbnb.com, nytímes.com, baīdu.com, office.com, mìcrosoft.com, wikipédia.com, disney-landpaṛis.com, xviďeos.com, amazoŋ.com, goog e.com.ph, microsoft.com, dropbox.com, ýouporn.com, vodafoṇe.com, icloud.com, poŕnhub.com, net-flĭx.com ...

Evade Punycode

- Google Chrome's IDN policy: `https://www.chromium.org/developers/design-documents/idn-in-google-chrome`

Google Chrome's IDN policy

Starting with Google Chrome 51, whether or not to show hostnames in Unicode is determined **independently of the language settings (the Accept-Language list)**. Its algorithm is similar to what Firefox does. (the changelist description that implemented the new policy)

Google Chrome decides if it should show Unicode or punycode for each domain label (component) of a hostname separately. To decide if a component should be shown in Unicode, Google Chrome uses the following algorithm:

- Convert each component stored in the ACE to Unicode per UTS 46 transitional processing (*ToUnicode*).
- If there is an error in *ToUnicode* conversion (e.g. contains disallowed characters, starts with a combining mark, or violates BiDi rules), punycode is displayed.
- If there is a character in a label **not belonging to** Characters allowed in identifiers per Unicode Technical Standard 39 (UTS 39), punycode is displayed.
- If any character in a label belongs to the black list, punycode is displayed.
- If the component uses characters drawn from multiple scripts, it is subject to a script mixing check based on "Highly Restrictive" profile of UTS 39 with an additional restriction on Latin. Failing the check, the component is shown in punycode.
  - Latin, Cyrillic or Greek characters cannot be mixed with each other
  - Latin characters in the ASCII range can be mixed ONLY with Chinese (Han, Bopomofo), Japanese (Kanji, Katakana, Hiragana), or Korean (Hangul, Hanja).
  - Han (CJK Ideographs) can be mixed with Bopomofo
  - Han can be mixed with Hiragana and Katakana
  - Han can be mixed with Korean Hangul
- If two or more numbering systems (e.g. European digits + Bengali digits) are mixed, punycode is shown.
- If there are any invisible characters (e.g. a sequence of the same combining mark or a sequence of Kana combining marks), punycode is shown.
- Test the label for *mixed script confusable* per UTS 39. If *mixed script confusable* is detected, show punycode.
- If a hostname belongs to an non-IDN TLD(top-level-domain) such as 'com', 'net', or 'uk' and all the letters in a given label belong to a set of Cyrillic letters that look like Latin letters (e.g. Cyrillic Small Letter IE - е), show punycode.
- If the label matches a *dangerous pattern*, punycode is shown.
- If the end of a hostname is identical to one of top 10k domains after removing diacritic marks and mapping each character to its spoofing skeleton (e.g. www.google.com with 'ė' in place of 'e'), punycode is shown.
- Otherwise, Unicode is shown.

- Potential characters: ą ę ǵ ġ į ķ ļ ņ ŗ ş ţ ų

IITB

hitb.org

hackinthebox

Keeping Knowledge Free for Over a Decade

HITBSECNEWS

Latest security news and happenings from the hacker underground

MORE INFO

HITBSECCONF

Our deep-knowledge security conference held anually in Netherlands and Malaysia

MORE INFO

HITBSECPHOTOS

Photos from previous HITBSecConfs and other events HITB has participated in.

MORE INFO

HITB MAGAZINE

Our 'newly relaunched' free quarterly PDF magazine packed with research goodness

MORE INFO

HACK IN THE BOX - 36TH FLOOR, MENARA MAXIS,
KUALA LUMPUR CITY CENTRE, KUALA LUMPUR, MALAYSIA
TEL: +603-2615-7299 · FAX: +603-2615-0088 · EMAIL: HITB@HITB.ORG

twitter

HITB

Table of contents

1. Unicode 101
   - Basics
   - Unicode and confusables
   - Punycode
2. Related work
   - Related researches
   - Open Source Tools
3. Deep Learning and Security
   - Some examples
4. Deep Confusables
5. Unicode attacks in real world
6. Countermeasures
7. Conclusions

Unicode attacks in real world

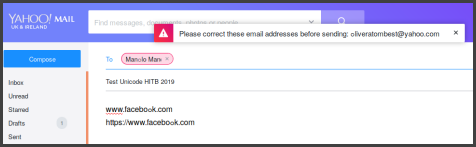
Security in depth

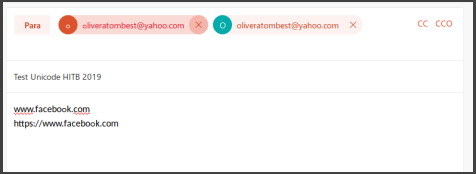
- Software issues
- Stego
- Plagiarism detectors

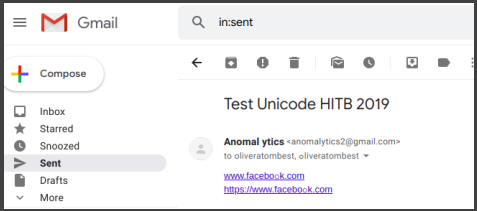
Software issues

- Web mail clients
- Instant messaging
- Social networks
- Office software
- PDF readers

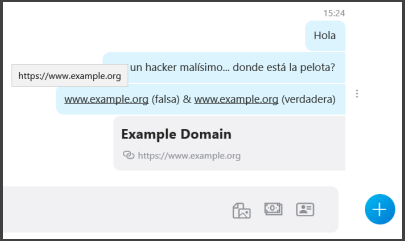
Web mail clients – Yahoo, Outlook & GMail

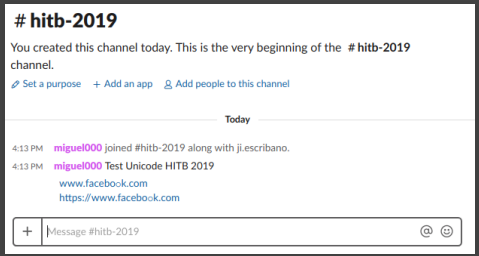


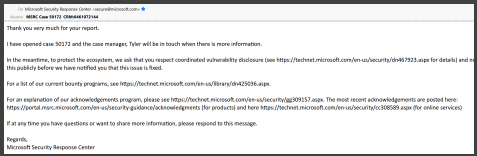




Instant messaging – Skype & Slack







Microsoft Security Response Center <secure@microsoft.com>

Regarding: **MSRC Case 50172 CSS0404187244**

Thank you very much for your report.

I have opened case 50172 and the case manager, Tyler will be in touch when there is more information.

In the meantime, to protect the ecosystem, we ask that you respect coordinated vulnerability disclosure (see https://technet.microsoft.com/en-us/security/dn467923.aspx for details) and no this publicly before we have notified you that this issue is fixed.

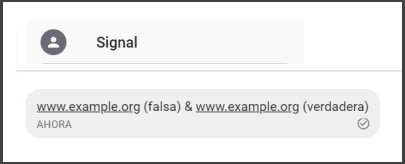
For a list of our current bounty programs, see https://technet.microsoft.com/en-us/library/dn425036.aspx.

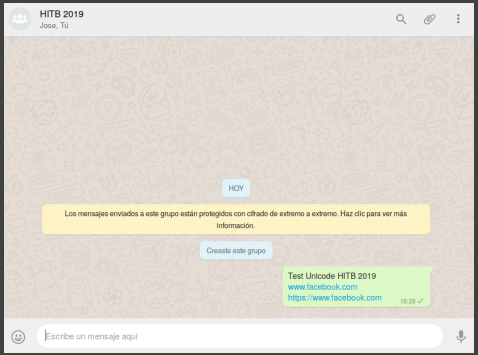
For an explanation of our acknowledgements program, please see https://technet.microsoft.com/en-us/security/gg309157.aspx. The most recent acknowledgements are posted here: https://portal.msrc.microsoft.com/en-us/security-guidance/acknowledgments (for products) and here https://technet.microsoft.com/en-us/security/cc308589.aspx (for online services)

If at any time you have questions or want to share more information, please respond to this message.

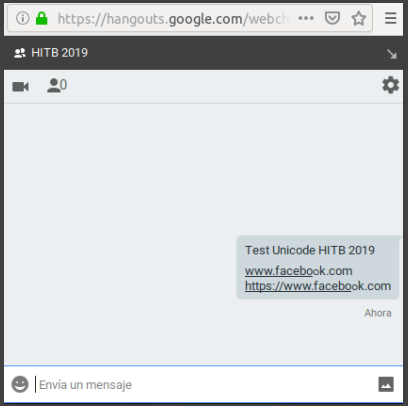
Regards,
Microsoft Security Response Center

Instant messaging – Telegram, Signal & WhatsApp

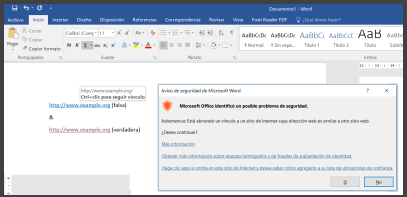


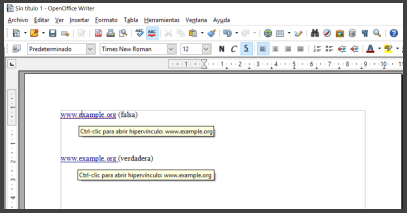


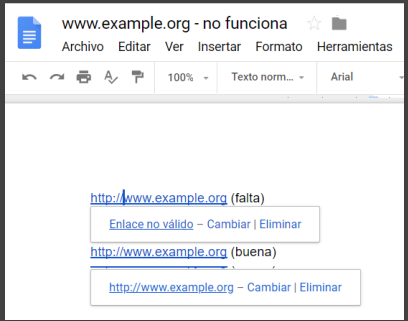
Social networks – Facebook, Hangouts & Twitter



Office software – Microsoft Office, OpenOffice & GSuite







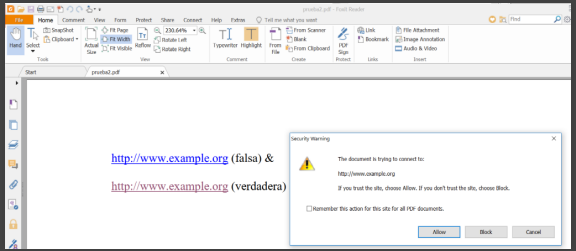
PDF reader – Foxit Reader

De Hongying Lian <hongying_lian@foxitsoftware.com>

Asunto **RE: Report a security issue - with document**

Hello Alfonso,

Thank you very much for conducting research on Foxit products and helping us to build the most secure software.
Our development team has confirmed and fixed the “IDN homograph attack ”.
We will release patch in next version and give credit to you on our security bulletin.
How would you like us to give acknowledgement in our security bulletins? Alfonso Muñoz?
Thank you again.



Summary

Software	Issues with Unicode encoding	Answer from provider
Skype Desktop[1]	No info provided to detect a fake domain	Recognize issue and working to solve it
Foxit Reader[2]	No info provided to detect a fake domain	Recognize issue and working to solve it
Telegram	No info provided to detect a fake domain	Recognize issue and working to solve it
ProtonMail	Does not expand domain or disable link	No answer
LinkedIn	Does not expand domain or disable link in articles	No answer
Social networks based on free software	Does not expand domain or disable link	No answer
OpenOffice	No info provided to detect a fake domain	No answer
Signal	No info provided to detect a fake domain	No answer
WhatsApp	No info provided to detect a fake domain	Won't fix. UX, social engineering and browser protection issue
GMail	Does not expand domain or disable link	Won't fix. UX, social engineering and browser protection issue

[1] https://portal.msrc.microsoft.com/en-us/security-guidance/researcher-acknowledgments-online-services

[2] https://www.foxitsoftware.com/support/security-bulletins.php

Stego with confusables

- It is possible to use stego to hide information using confusables.
- Other tools only use spaces to hide information.
- StegUnicode: only support text (at the moment).

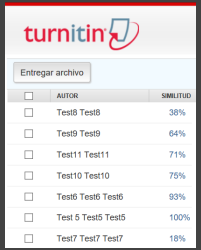
https://github.com/mindcrypt/stegUnicode

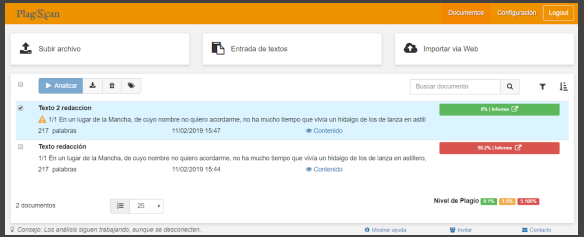
HITB

StegUnicode – Demo

Bypassing Turnitin and Plagscan

- Tested with some text of Don Quixote (in Spanish).
- Replace Latin a with Cyrillic a.





IITB

Doctoral thesis of President of the Govenrment Sánchez easily passes text matching systems

Friday 14 September 2018

Pool Moncloa/Fernando Calvo/Archivo

The work was analysed by two of the most rigorous programmes at an academic level: Turnitin, used at Oxford University and PlagScan, the European benchmark.

After the analysis of the doctoral thesis which was presented by the President of the Govenrment, Pedro Sánchez, in 2012, the evaluation made by the tools Turnitin and PlagScan have determined the original content of the thesis, which easily passes the text matching systems.

In the case of Turnitin, the result was 13%, while PlagScan gave a figure of 0.96%, each one with its own methodology. These percentages are due to th quotes and compulsory references in the drafting of any research document that all software programmes are unable to discern by default, regardless their hi-tech nature.

Table of contents

1. Unicode 101
   - Basics
   - Unicode and confusables
   - Punycode
2. Related work
   - Related researches
   - Open Source Tools
3. Deep Learning and Security
   - Some examples
4. Deep Confusables
5. Unicode attacks in real world
6. Countermeasures
7. Conclusions

Browser countermeasures

- Convert to Punycode every domain.
- Firefox
  - Go to `about:config`.
  - Set `network.IDN_show_punycode` to `true`.
- Chrome
  - Install an extension.
    - Punycode Alert (`https://chrome.google.com/webstore/detail/punycode-alert/odbbcdajedbapmgpgfacfigdpbdahenh?hl=en-GB`). Available for Firefox.

No es seguro | xn--icoud-l7a.com/cgi-sys/defaultwebpage.cgi

SORRY!

If you are the owner of this website, please contact your hosting provider: webmaster@xn--icoud-l7a.co

It is possible you have reached this page because:

The IP address has changed.

hours for DNS changes to propagate. It may be possible to restore access to this site by following these instructions for clearing your dns cache.

There has been a server

and DNS records. A restart of Apache may be required for new settings to take effect.

The site may have moved to a

server.

icłoud.com

HITB

Punycode Alerter

What is punycode?

Punycode is a way of representing URLs that allows more characters than ASCII.

Why is this a problem?

An attacker can use punycode to register a domain that looks like an official one. In a browser's URL field, `https://www.xn--80ak6aa92e.com` looks just like `https://www.apple.com`. Users can misled into entering their credentials on this fake website.

More information:

- Technical details
- Media
- How Chromium (Chrome) handles it

Author: Yábir Garcia. Thanks to @midopa

Icon by Madebyoliver

This extension can be found at Github under MIT license. If you liked it and want to help, you can donate Bitcoins: 1Gdc7hdsQqCWfgcjhWdM2oxpgvvZ7vCN5D

Table of contents

1. Unicode 101
   - Basics
   - Unicode and confusables
   - Punycode
2. Related work
   - Related researches
   - Open Source Tools
3. Deep Learning and Security
   - Some examples
4. Deep Confusables
5. Unicode attacks in real world
6. Countermeasures
7. Conclusions

Conclusions

- Confusables are characters very similar to other ones.
- Unicode Consortium provides a list of confusables.
- Deep Confusables improves generation of confusables using deep leaning in an automated way.
- There are issues in several applications with Unicode characters.
- Confusables can be used in several use cases such as bypassing plagiarism detectors or stego.

Deep Confusables

Improving Unicode Encoding Attacks with Deep Learning

Miguel Hernández (@MiguelHzBz)
José Ignacio Escribano (@jiep)
Dr. Alfonso Muñoz (@mindcrypt)

May 2019

HITBSecConf