

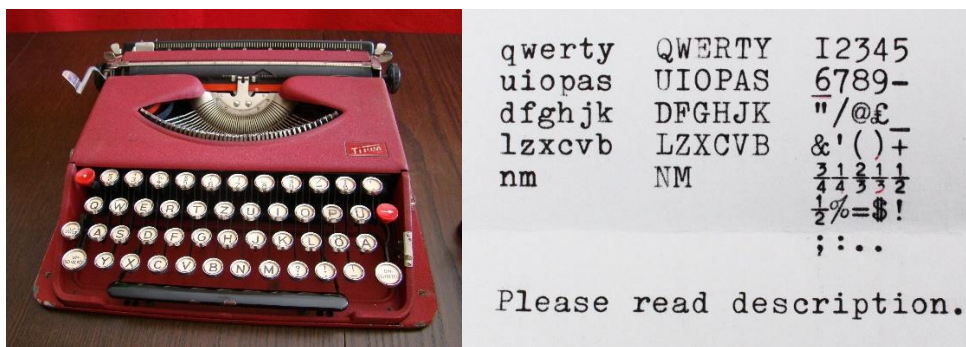
Unicode 编码视觉欺骗攻击深度解析

xisigr 2022/9/21

一、前言

2022 年 9 月 13 日, Unicode 15.0 正式版发布。在 Unicode15.0 其中增加了 4489 个字符, 总共字符数量达到了 149,186 个。这些新增内容包括 2 个新脚本, 总共脚本数量达到 161 个, 以及 20 个新的表情符号字符。同时几个重要的 Unicode 规范也随着 15.0 版本进行了更新, 这其中就包括 Unicode 安全机制(UTS # 39)这个规范, 它意在减少因 Unicode 字符视觉欺骗带来的同形异意攻击 (Homoglyph Attack) 。

同形异意攻击 (Homoglyph Attack) 是非常古老的一种视觉欺骗攻击方式。在机械打字机时代, 很多打字机为了简化设计和降低制造及维护成本, 键盘上没有单独的 1 和 0。打字员会使用小写字母 l 和大写字母 i 来代替数字 1, 使用大写字母 O 来代替数字 0。当这些相同的打字员在 70 年代和 80 年代初转变为计算机键盘操作员时, 他们的旧键盘习惯在他们的职业中继续存在, 并成为极大混乱的源泉。这应该是视觉混淆、同形异意攻击集中爆发的一个时期。

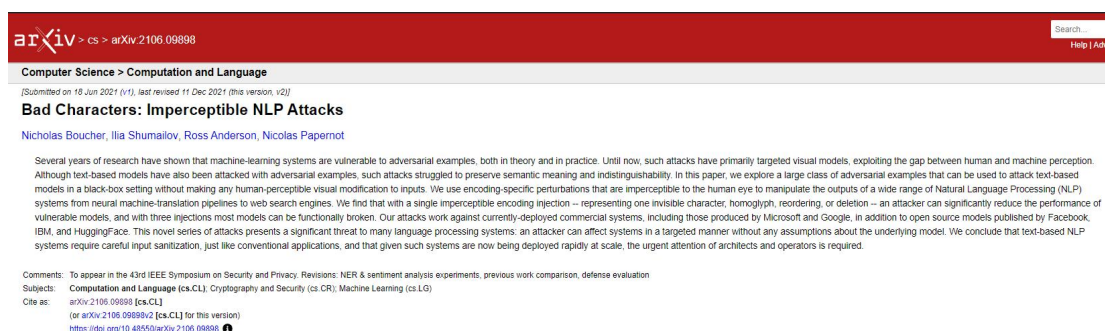


在此之后, 打字机被文字处理器所代替, 信息化时代逐步到来, 字符编码也开始由 ASCII 字符集逐渐扩充到 Unicode 字符集。我们开始使用浏览器或其他应用客户端来呈现文本, 那些在某些语境中不适合使用同形字书写 URL, 公式, 源代码, ID 等等, 其相似的外观继

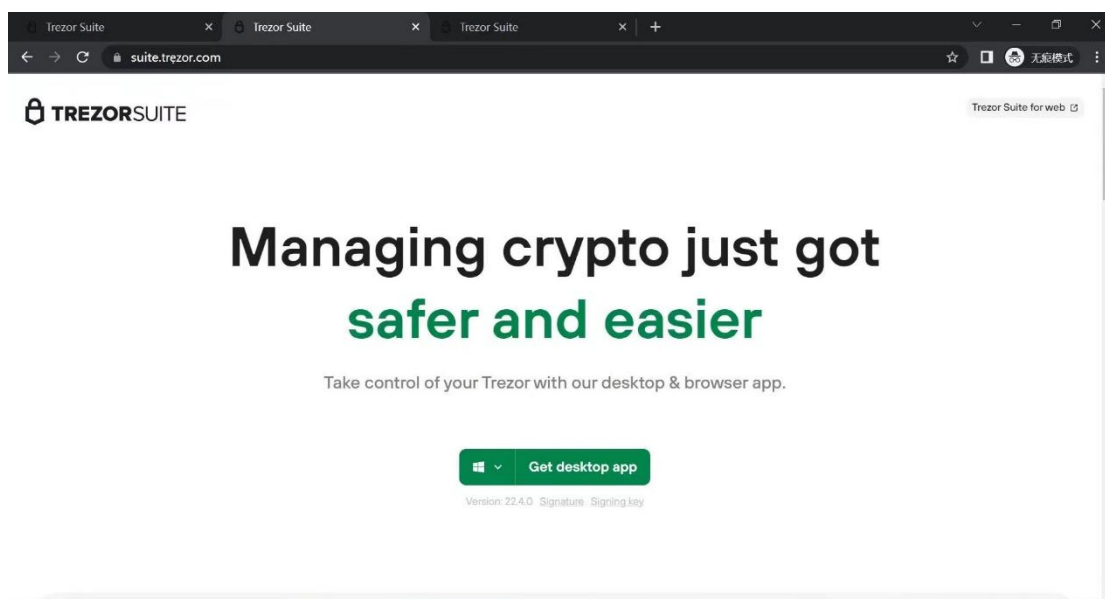
续使得用户可能在视觉上无法区分。

Unicode 视觉欺骗取决于视觉上可以混淆的字符串：两个 Unicode 字符串外观上非常相似，在通常的屏幕分辨率下，它们以小尺寸的普通字体出现，很容易让人们误认为是另一个。视觉混淆没有明显的规则：当尺寸足够小时，许多字符看起来像其他字符。‘屏幕分辨率下的小尺寸’是指大多数脚本使用 9-12 像素的字体。易混淆性还取决于字体的风格：对于传统的希伯来字体，许多字符只能通过细微的差别来区分，而这些细微差别可能在小尺寸时丢失。在某些情况下，字符序列也可用于欺骗：例如，“rn”（“r”后跟“n”）在许多 sans-serif 字体中与“m”在视觉上混淆。

近些年来，因 Unicode 编码欺骗发生了很多恶意攻击事件，人为或编译器或 AI 都可能因 Unicode 欺骗产生错误的判断和解析。例如 2021 年有研究人员在谷歌等商业系统中使用 Unicode 的这些特殊字符对 NLP 模型进行对抗攻击。他们通过一些不可察觉的编码注入--比如一个不可见字符、同形符、重新排序或删除的操作字符，可以显著降低一些模型的性能，大多数模型都可能在功能上失灵（<https://arxiv.org/abs/2106.09898>）。

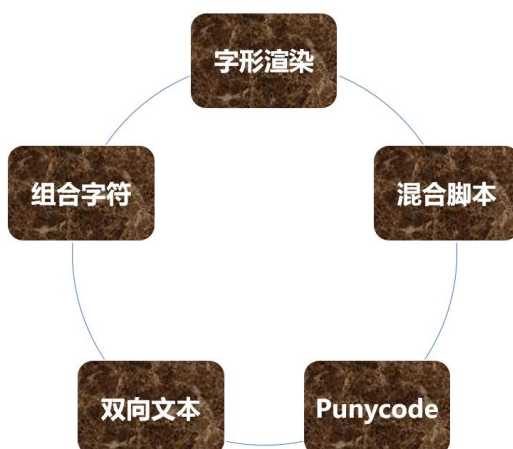


2022 年 Trezor 这款知名的硬件钱包出现了大量的网络钓鱼网站，钓鱼链接 <https://suite.trezor.com>。这个钓鱼链接和真实 Trezor 官方网站 trezor.io 极为相似。（此案例来源于“区块链黑暗森林自救手册”）



虽然基于客户端的尤其是浏览器端的抵御视觉欺骗的防御措施一直在完善,但并没有办法完全封堵住这种攻击。第一,视觉欺骗是一种很难被完全消亡的攻击,因为在足够小的尺寸情况下人自身的生理视觉体系也是无法分辨的;第二 Unicode 字符集非常庞大,并且在不断的增加;第三,需要标准组织、浏览器开发商、域名注册商等多方共同协作去完成。

本文主要研究当下 Unicode 中因字形渲染、混合脚本、PunyCode、双向文本、组合字符引起的视觉欺骗问题,结合作者发现的不同类型的 Unicode 视觉欺骗漏洞,分享漏洞挖掘过程方法并给出一些防御的思路。



二、 字形渲染带来的安全风险

当字体或渲染引擎对字符或字符序列的支持不足时,就使得本应该在视觉上可以区分的字符或字符序列出现新的问题--视觉混淆。尤其是这些字符被做为关键的信息出现时,比如在浏览器重要的安全指示器地址栏中,这种视觉欺骗的危害就会显而易见。浏览器厂商对于地址栏上的 IDN 欺骗问题,一直在做积极的防御,通常是在浏览器里维护一个重要域名列表,如果一个域名和列表中的域名同形异议,则将其转化为 Punycode 编码显示。

渲染支持不足,直接导致每一个字符的字形含义往往超出我们的预期,这种不确定性使得视觉欺骗随时都可能发生。那么字形到底是什么,它和字型、字符、字体有什么关系,包括 Unicode 中越来越火的 Emoji 表情它是字符吗?其中一些基本的名词概念非专业人士经常混淆,我们先简要了解下。

英文	中文
Typeface	字体
Font	字型
Glyph	字形
Character	字符
Character set	字符集
Emoji	表情符号

字体 (英语: Typeface) 指的是一组字符的设计,通常包括字母、一组数字和一组标点符号。也常包括表意字符以及制图符号。每个字体都是字形的集合,比如宋体、黑体等。

有些字体技术非常强大,比如 TrueType / OpenType, 它们可以根据分辨率、系

统平台、语言等来选择显示最佳的形状。但是，它也可以用于安全攻击，因为它足够强大，可以在打印时将屏幕上的“\$ 1 00.00”外观更改为“\$ 2 00.00”。此外，层叠样式表(CSS)可以更改为不同的字体，用于打印与屏幕显示，这可以使用更多可混淆字体。这些问题并非特定于 Unicode。为了降低此类漏洞利用的风险，程序员和用户应该只允许可信任的字体。

字型（英语：font；传统英式英语：fount）是指印刷行业中某一整套具有同样样式和尺码的字形，例如一整套用于内文的宋体 5 号字、一整套用于标题的 10 号字就叫一套字型。电脑早期用点阵字，仍然有字型概念，同样一套风格如中易宋体，一套字型是指一整套 15×16 像素或一整套 24×24 像素的字。矢量字型出现后，同一套风格字型已不用制作不同像素字型，只需制作一套即可随意缩放，“字型”与“字体”之间的界限开始模糊。一般的英语使用者同样分不清“字型”（Font）与“字体”（Typeface）的分别。

字形（英语：glyph），又称字图或书形，是指字的形体。中华人民共和国国家标准 GB/T 16964《信息技术·字型信息交换》中定义字形为“一个可以辨认的抽象的图形符号，它不依赖于任何特定的设计”。在语言学中，字（character）是语意的最基本单位，即语素；字形是指为了表达这个意义的具体表达。同一字可以有不同的字形，而不影响其表达的意思，例如拉丁字母第一个字母可以写作 a 或 α，汉字中的“強 / 强”、“戶 / 户 / 戸”。



在复杂的脚本（如阿拉伯语和南亚语脚本）中，字符可能会根据周围的字符更改形状。

(1) 字形可以随周围环境所改变

3 个 arabic letter heh (U+0647)组合在一起

ه + ه + ه = ههه

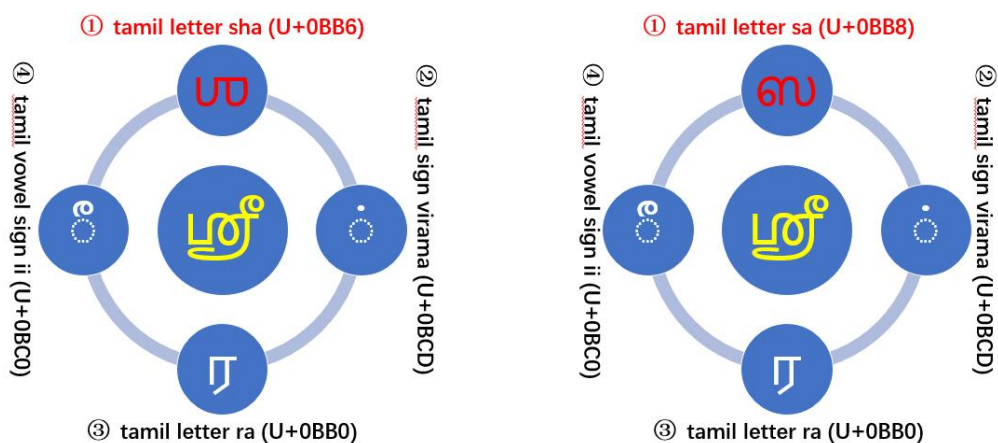
(2) 多个字符可以产生一个字形

f = latin small letter f (U+0066)

i = latin small letter i (U+0069)

f + i = fi

我们来看一个和视觉有关的，当多个不同字符组合后可能视觉外观是相同的。例如字符 U + 0BB6 SHA 和 U + 0BB8 SA 通常非常不同。但这两个组合的字符序列，视觉感官上完全相同。底层的二进制是不同的。

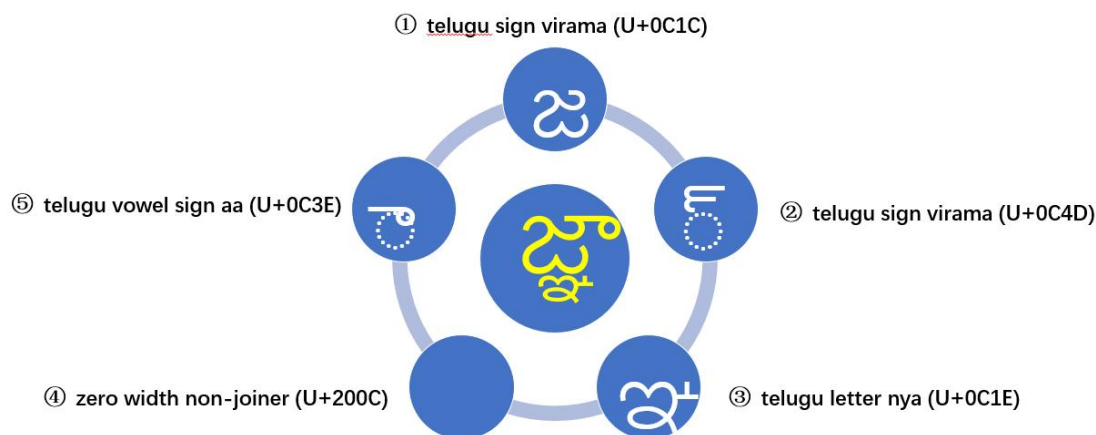


其实多字符组合不仅仅是视觉欺骗，有时这种多字符组合也可以导致系统内存崩溃。

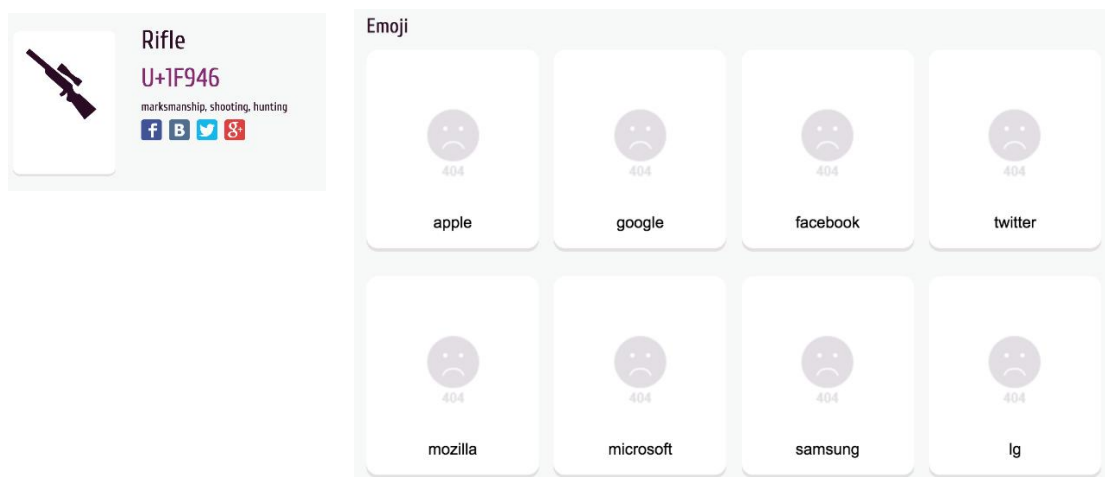
CVE-2018-4124 这个漏洞可导致 macOS High Sierra 10.13.3 在处理恶意制作的字符串时致堆损坏。☞的原始序列是 U+0C1C U+0C4D U+0C1E U+200C U+0C3E，这是一个泰卢固语字符序列：辅音 ja (జ)、virama (్)、辅音 nya (న్)、零-width 非连接符和元音 aa

(☹)。当 macOS 在处理𑖦这个字符序列时，即可导致系统崩溃。

CVE-2018-4124



表情符号（英文：Emoji）是象形文字（图形符号），通常以彩色卡通形式呈现并在文本中内联使用。它们代表面部，天气，车辆和建筑物，食物和饮料，动物和植物，或代表情感，感觉或活动的图标。Emoji 已经无处不在，它正在成为跨越不同文化的所有人通用的语言。但不同人对 Emoji 的使用理解以及各个操作系统碎片化的支持，也使得带来了一些问题。比如：短期内你也不会看到步枪的表情符号，U+1F946。Unicode 组织包括苹果和微软都反对加入步枪符号。手枪和其他武器的 emojis 已经使人们陷入了法律的困扰。之前一家法国法院裁定手枪表情符号可能构成死亡威胁，将一名把枪的表情发给前女友的男子判处三个月监禁。



在浏览器地址栏中直接渲染 U+1F512 这个编码也是相当危险的。因为它和 HTTPS 安全小锁外观上非常相似。攻击者可以对其进行安全小锁的伪造。U+1F512 编码曾在 Chrome/Firefox 浏览器中出现过这样的安全问题。



lock (U+1F512)

三、 混合脚本带来的安全风险

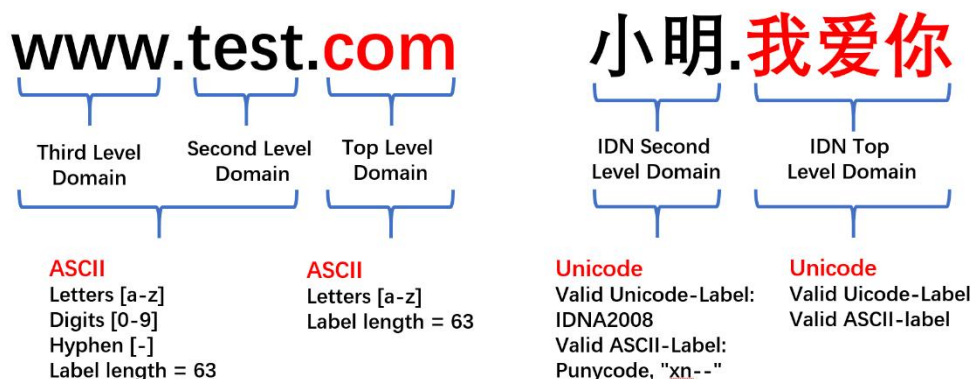
混合脚本是有很多合法的用途的，例如Ωmega。但视觉上容易混淆的字符通常不会放在一个脚本中。在视觉上容易混淆的字符为欺骗提供了许多机会，例如下面这两个域名，希腊小写字母 Omicron 和拉丁文 o 外观是非常相似的。

String	UTF-16	Punycode
top.com	0074 03BF 0070 002E 0063 006F 006D	xn--tp-jbc.com
top.com	0074 006F 0070 002E 0063 006F 006D	top.com

很久之前，域名只允许包含拉丁字母 A-Z，数字和一些其他字符（ASCII 字符集）。之后大家发现，仅仅支持 ASCII 码的域名可能是有问题的，因为世界上还有很多非拉丁文语系的国家和地区，他们也渴望在域名中使用自己的语言符号。后来经过多次提案讨论，在 2003 年发布了国际化域名的规范[rfc3490]，它允许大多数的 Unicode 在域名中使用，普遍将这个规范称为 IDNA2003。之后在 2010 年批准发布了对 IDNA2003 的修订版[rfc5895]，称这个修订版为 IDNA2008。但 IDNA2003 和 IDNA2008 并没有有效的解决国际化域名中的某些问题。随后，Unicode 联盟发布了[UTS-46]解决了某些兼容性的问题。

在 IDNA 中使用 PunyCode 算法来实现非 ASCII 域名到 ASCII 域名的转换。PunyCode 算法可以将任何一个非 ASCII 的 Unicode 字符串唯一映射为一个仅使用英文字母、数字和连字符的字符串，编码的域名在前面都加上了 xn-- 来表明这是一个 PunyCode 编码。

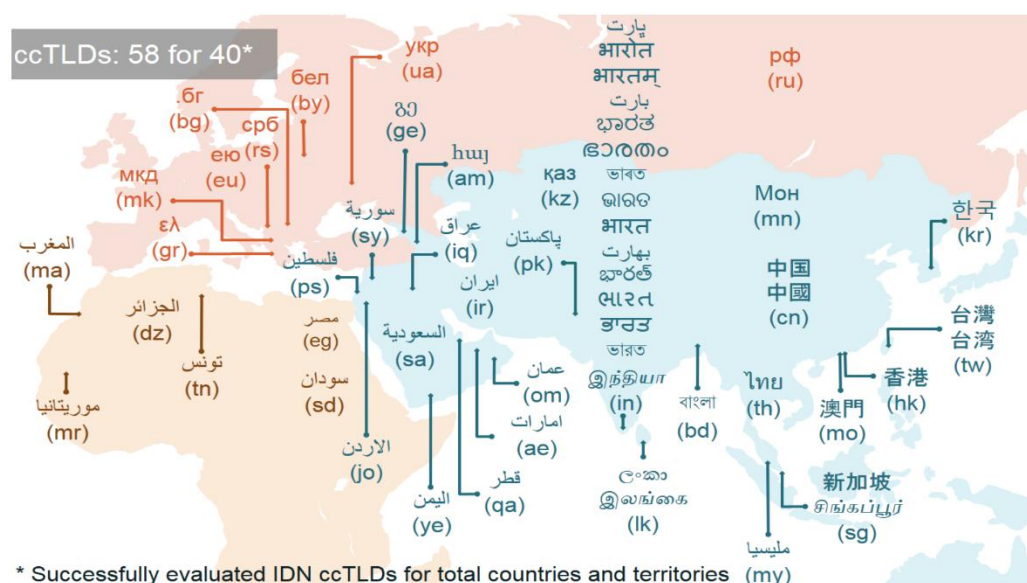
ASCII Domain names vs. Internationalized Domain Names (IDNs)



国际化域名 (IDN) 是在 Unicode 中定义的任何字符集或脚本中注册的二级或三级域名或 Web 地址。直到 2009 年底之前, 顶级域名仅限于拉丁字母 a-z, 之后随着 Web 全球化发展, IDN TLDs 也开始逐渐推广和普及, 这加速了全球化进展的同时, 也带来了一些安全风险, 这点在后面我们会谈到。

IDN TLD

- gTLD
 - com, org, net, edu, gov, mil.....
 - شبكة(网络), онлайн(在线), グーグル(谷歌), 游戏.....(IDN gTLD)
- ccTLD
 - cn, jp, nz, hr, be, cc.....
 - موريتانيا(毛里塔尼亚), 新加坡, 한국(韩国), السودان(苏丹).....(IDN ccTLD)



在前文中我们已经了解到, Unicode15.0 中的字符总量 149,186 个, 脚本数 161 个。这些脚本中大部分都可以被用于域名注册, 并且数量还在增加中。以顶级域名 (TLD) COM 为例, Verisign 制定了 IDN 注册的策略, 规定了允许和禁止的代码点。并制定了 IETF 标准、对特定语言的限制、对脚本混淆的限制、ICANN 受限 Unicode 代码、特殊字符这五条验证

规则，遵循这五条规则的 IDN 被认为是有效的注册。

在这五条 IDN 注册规则中，我们会重点关注“脚本混淆的限制”这条规则，因为这对发现 IDN 上的视觉欺骗问题有帮助意义。

【对脚本混淆的限制】

Verisign 不允许使用混合的 Unicode 脚本进行注册。如果 IDN 中包含二个或多个 Unicode 脚本代码，将拒绝注册。例如拉丁文脚本中的字符不能和任何西里尔字符在同一个 IDN 中使用。IDN 中的所有代码必须来自同一个 Unicode 脚本。这样做是为了避免混淆的代码出现在同一个 IDN 中。

下表列出了允许使用的 Unicode 脚本。

Unicode Scripts And Associated Code Points

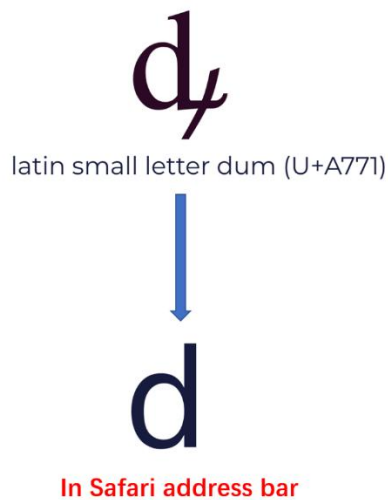
Arabic	Georgian	Latin	Rejang
Armenian	Glagolitic	Lepcha	Runic
Avestan	Greek	Limbu	Samaritan
Balinese	Gujarati	Lisu	Saurashtra
Bamum	Gurmukhi	Lycian	Sinhala
Batak	Han	Lydian	Sundanese
Bengali	Hangul	Malayalam	Syloti Nagri
Bopomofo	Hanunoo	Mandaic	Syriac
Brahmi	Hebrew	Meetei Mayek	Tagalog
Buginese	Hiragana	Mongolian	Tagbanwa
Buhid	Imperial Aramaic	Myanmar	Tai Le
Canadian Aboriginal	Inscriptional Pahlavi	New Tai Lue	Tai Tham
Carian	Inscriptional Parthian	Nko	Tai Viet
Cham	Javanese	Ogham	Tamil
Cherokee	Kaithi	Ol Chiki	Telugu
Coptic	Kannada	Old Persian	Thaana
Cuneiform	Katakana	Old South Arabian	Thai
Cyrillic	Kayah Li	Old Turkic	Tibetan
Devanagari	Kharoshthi	Oriya	Tifinagh
Egyptian Hieroglyphs	Khmer	Phags Pa	Vai
Ethiopic	Lao	Phoenician	Yi

列举我之前发现的一个漏洞，[CVE-2018-4277] Spoof All Domains Containing 'd' in Apple Products 。我在研究中发现，在苹果产品中编码 latin small letter d (U+A771)渲染的字形和 latin small letter d (U+0064)极为相似。从 Unicode 中(U+A771)的字形标准可以发现 (<http://www.unicode.org/charts/PDF/UA720.pdf>)，d 后面应该

还有一个小撇，但是在苹果产品中把这个完全忽略掉了。

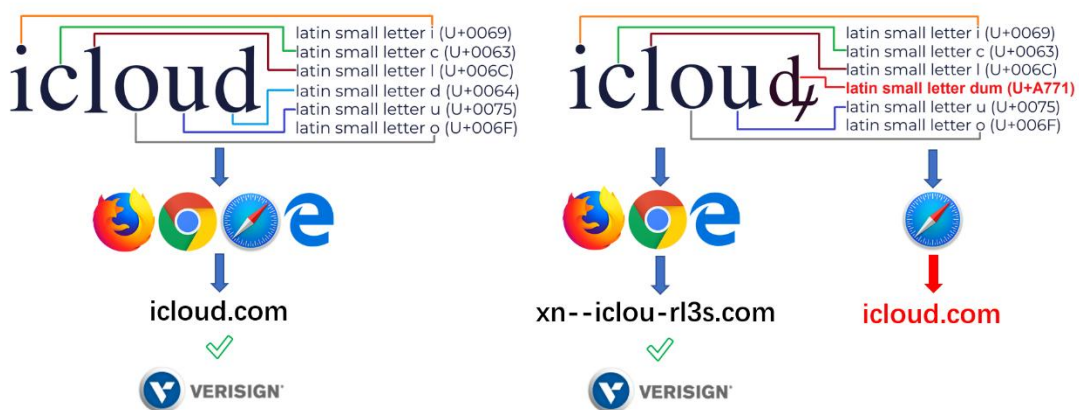
	Latin Extended-D															
	A720	A721	A722	A723	A724	A725	A726	A727	A728	A729	A72A	A72B	A72C	A72D	A72E	A72F
0	ƒ	F	K	P	W	9	7	N	G	X						
1	ƒ	s	k	p	w	ɔ	l	n	g	L						
2	3	AA	K	P	3	l	ɾ	C	K	J						
3	3	aa	k	p	3	m	ɾ	e	k	X						
4	c	AO	K	P	ɔ	ɾ	ɾ	ɾ	N	B						
5	c	ao	k	p	ɔ	ɾ	ɾ	ɾ	n	β						
6	H	AJ	L	Q	P	R	C	B	R	W						
7	h	ai	l	q	p	t	ɾ	ɾ	ɾ	ɾ						ɾ
8	B	A	L	Q	V	ɔ	ɾ	f	s							h
9	t	a	i	q	ɾ	ɾ	:	f	s							œ
A	E	A	Θ	ɾ	3	ɾ	=	ɾ	H							W
B	E	æ	θ	ɾ	3	ɾ	ɾ	a	3							ɾ
C	4	A	Θ	ɾ	ɾ	ɾ	ɾ	ɾ	g							ɾ
D	4	ɾ	σ	ɾ	ɾ	ɾ	ɾ	ɾ	L							W
E	4	ɾ	∞	W	9	ɾ	ɾ	ɾ	ɾ							I
F	4	ɾ	∞	ɾ	9	ɾ	ɾ	ɾ	ɾ							W

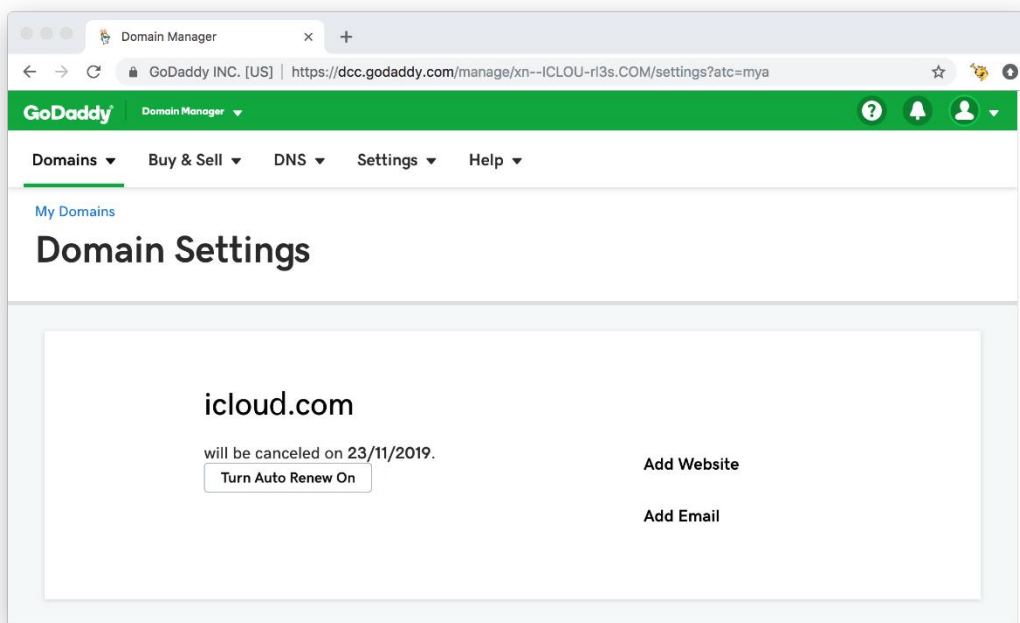
The Unicode Standard 10.0, Copyright © 1991-2017 Unicode, Inc. All rights reserved.



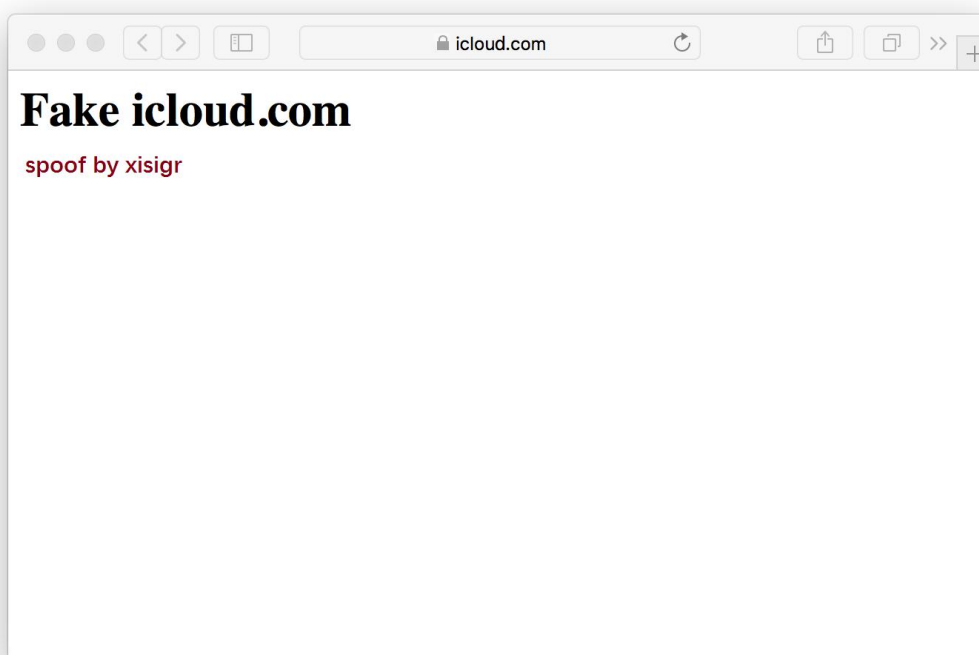
接下来，我去注册了一个真实的域名，使这个 IDN Spoof 可以正常运行。我们知道在 Verisign 制定的 IDN 注册的规则中，不允许使用混合的 Unicode 脚本进行注册。如果 IDN 中包含二个或多个 Unicode 脚本代码，将拒绝注册。而 (U+A771) 也是属于 Latin，应该是符合域名注册商规则的。于是域名成功注册成功了。

Latin: icloud.com VS Latin Extended-D: icloud.com





我又注册了一个 SSL 证书，使这个 IDN Spoof 看的会更加真实完美。效果如下，Safari 并没有将这个域名转化为 punycode 显示，所以我们成功了。



到这里我们确定了整个欺骗流程是完全可行的，那么攻击者可以伪造域名中有 d 的所有域名。在 google 统计的 Top 10k 域名中，大约有超过 25% 的网站域名中有 d 这个字符。这

些网站的域名都可以被伪造。

https://chromium.googlesource.com/chromium/src/+master/components/url_for_matter/top_domains/alexa_domains.list

- linkedin.com
- baidu.com
- jd.com
- adobe.com
- wordpress.com
- dropbox.com
- godaddy.com
- reddit.com
-

【苹果修复的补丁】

```
1  LayoutTests/fast/url/host-expected.txt View
@@ -47,6 +47,7 @@ PASS canonicalize('http://@google.com/') is 'http://google.com/'
47 47 PASS canonicalize('http://quip-apple.com/') is 'http://xn--quipapple-y79d.com/'
48 48 PASS canonicalize('http://quip-apple.com/') is 'http://xn--quipapple-y79d.com/'
49 49 PASS canonicalize('http://quip-apple.com/') is 'http://xn--quipapple-tf4e.com/'
50 + PASS canonicalize('http://icloud.com/') is 'http://xn--iclou-r13s.com/'
50 51 PASS successfullyParsed is true
51 52
52 53 TEST COMPLETE

3  LayoutTests/fast/url/host.html View
@@ -87,7 +87,8 @@
87 87 ["@google.com", "google.com"],
88 88 ["quip\u2010apple.com", "xn--quipapple-y79d.com"],
89 89 ["quip\u2011apple.com", "xn--quipapple-y79d.com"],
90 - ["quip\u2212apple.com", "xn--quipapple-tf4e.com"]
90 + ["quip\u2212apple.com", "xn--quipapple-tf4e.com"],
91 + ["iclou\uA771.com", "xn--iclou-r13s.com"]
91 92 ];
92 93
93 94 for (var i = 0; i < cases.length; ++i) {
```

【受影响产品】

watchOS 4.3.2 <https://support.apple.com/zh-cn/HT208935>

iOS 11.4.1 <https://support.apple.com/zh-cn/HT208938>

tvOS 11.4.1 <https://support.apple.com/zh-cn/HT208936>

macOS High Sierra 10.13.5 <https://support.apple.com/zh-cn/HT208937>

四、 双向文本带来的安全风险

某些字符（例如阿拉伯语和希伯来语脚本中使用的字符）具有固有的从右到左的书写方向。当这些字符与从左到右显示的其他脚本或符号集的字符混合时，生成的文本称为双向（缩写为 bidi）。文档的内存表示（逻辑顺序）与双向文本的显示外观（可视顺序）之间的关系由 UAX #9: Unicode 双向算法 [UAX9]管理。

由于某些字符具有弱或中性的方向性，而不是强左向右或从右到左，因此 Unicode 双向算法使用一组精确的规则来确定最终的视觉呈现。然而，任意文本序列的呈现，可能导致文本序列无法清晰地被阅读，或者可能在视觉上混淆。

在一个 URL 中，经常会遇到多种方向性（弱性、中性、强性）的字符同时存在的情况。虽然 Unicode 双向算法使用一组精确的规则来确定最终的视觉呈现，但是多种方向性的文本序列在呈现时，还是可能导致文本序列无法清晰地被阅读，或者可能在视觉上混淆。

LTR vs RTL

- URL-LTR
 - Subdomain: hi
 - Domain: google
 - TLD: com
 - Path: search
- http://hi.google.com/search
- URL-RTL(hebrew)
 - Subdomain: ה
 - Domain: ג
 - TLD: ל
 - Path: מ
- 渲染: http://מ/ל.ג.ה

Chrome 之前出过这样一个漏洞，在 Chrome 里访问：

http://127.0.0.1/%D8%A7/example.org

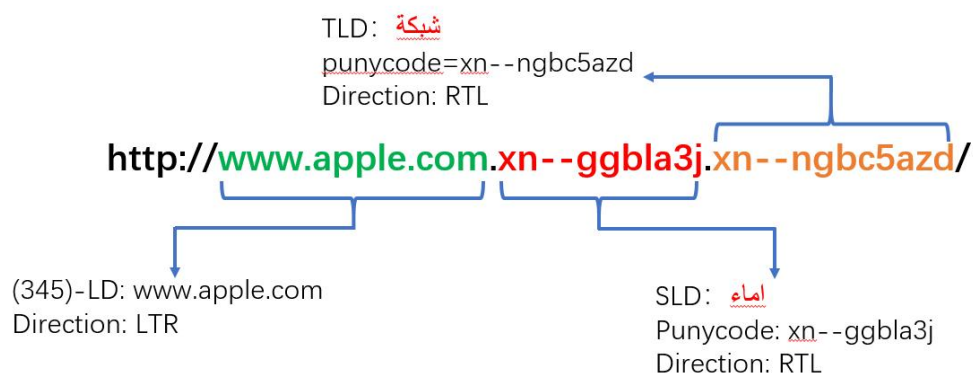
在地址栏中实际渲染为：

http://example.org/127.0.0.1

CVE-2018-4205 是我之前发现的一个漏洞，利用了 RTL 和空白符导致了 URL 地址栏欺骗。下面就以这个漏洞为例进行讲解。

(1) 构造 POC-1

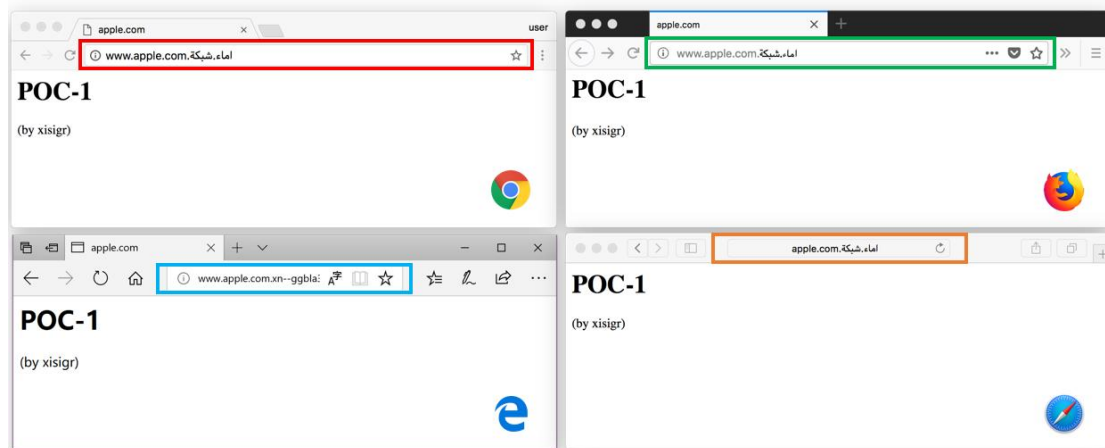
POC-1



访问 `http://www.apple.com.xn--ggbla3j.xn--ngbc5azd/`。可以看到

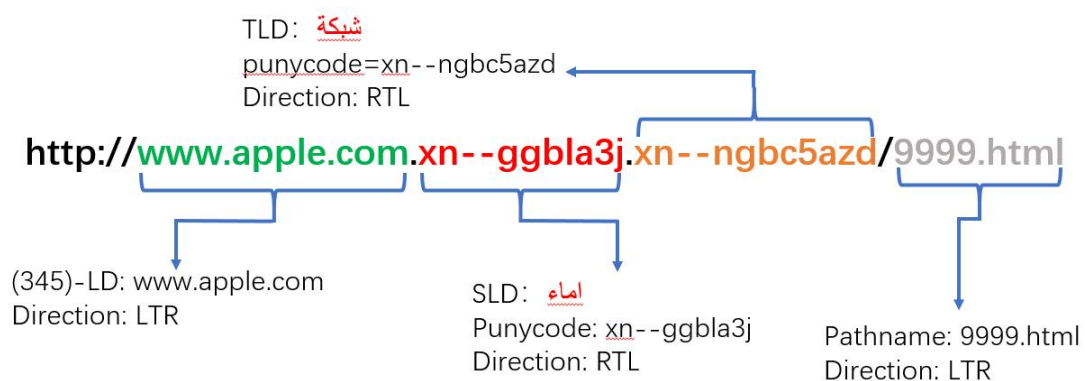
Chrome/Firefox/Safari 三个浏览器地址栏，显示都是一样的。Edge 显示 punycode。

POC-1



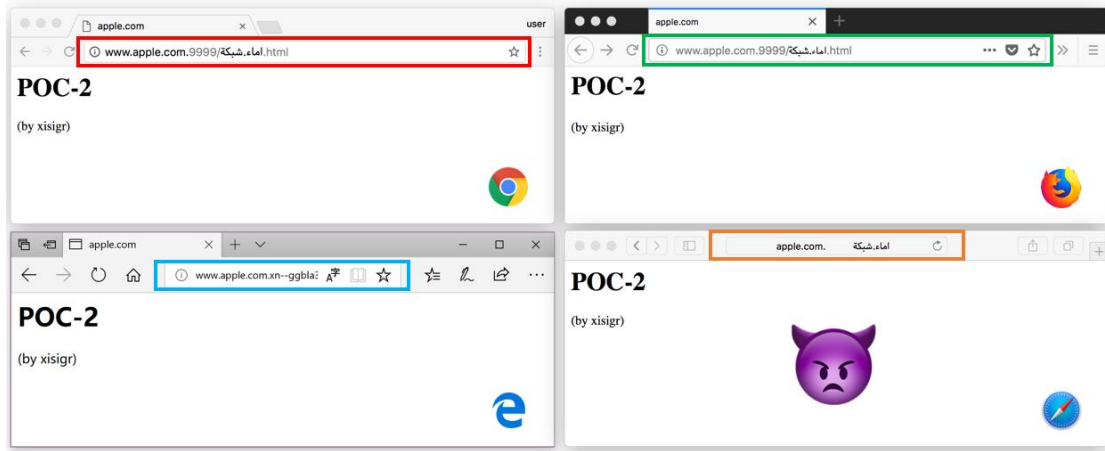
(2) 构造 POC-2

POC-2



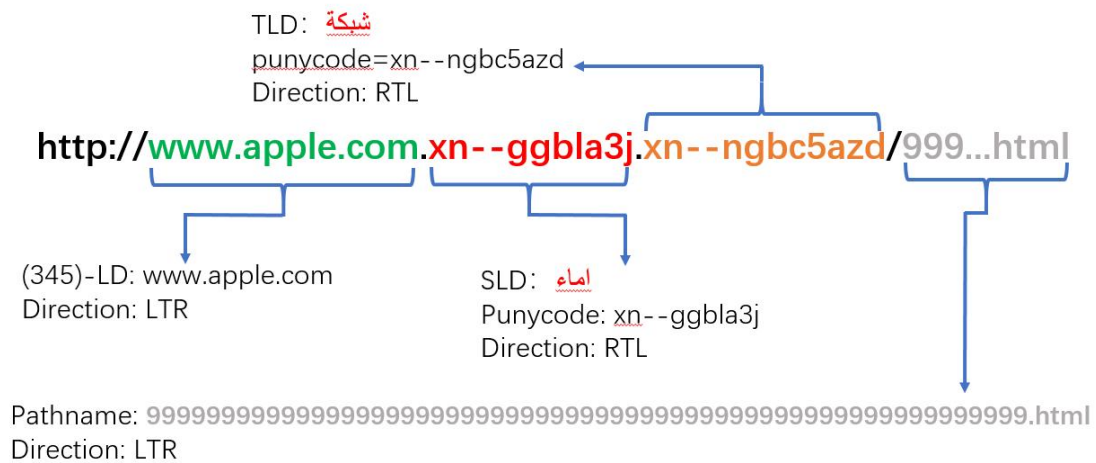
在四个浏览器中访问 POC-2，我们此时已经发现了问题。在 Safari 中出现了 RTL 和空白字符。

POC-2

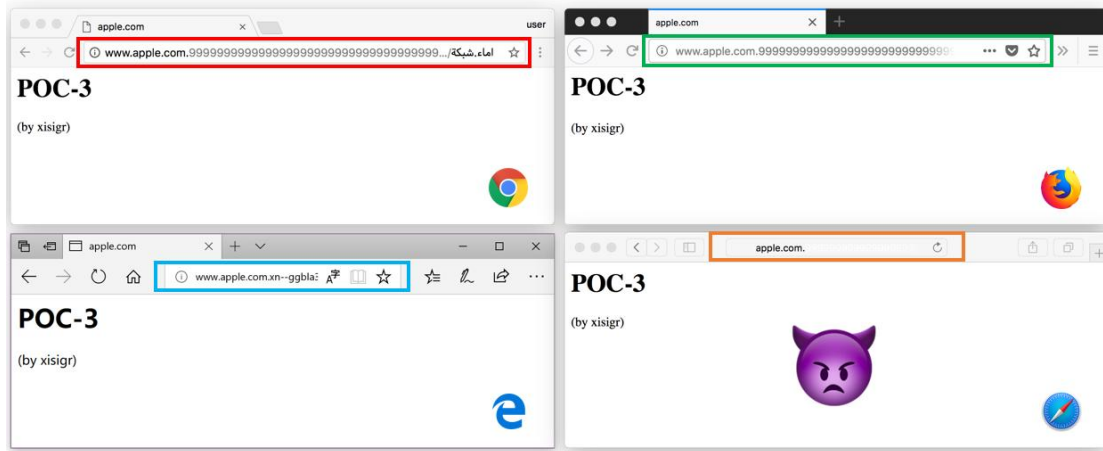


(3) 构造 POC-3

POC-3



POC-3

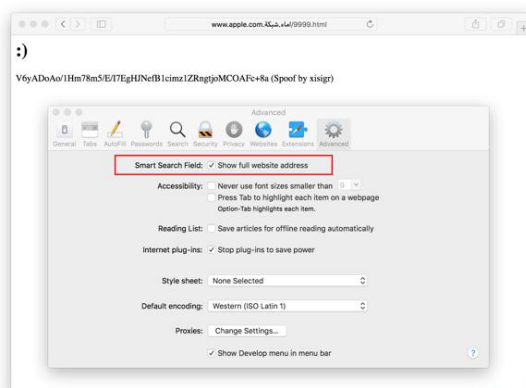


(4) 根因分析

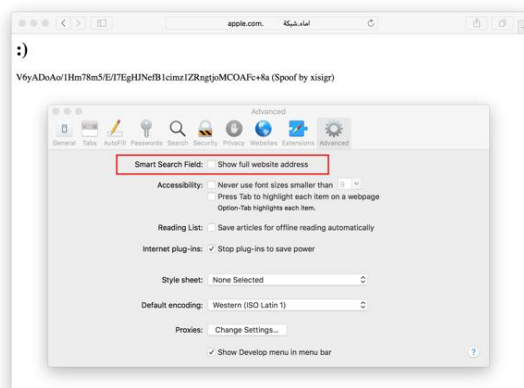
针对 POC-3，在 Safari 中的地址栏欺骗已经比较完美。在 Safari 中当设置为只显示域名不显示完整 URL 时，意味着 pathname 部分是去掉的不会显示的。这个字符串 `http://www.apple.com.xn--ggbla3j.xn--ngbc5azd/999...html`，如果按常理显示就是这样。但 `اماء شبكة` 是 RTL，使字符串产生了乱序。`http://www.apple.com.9999/اماء شبكة.html`，9999 pathname 部分和 `اماء شبكة` 调换了位置。字符串变为 `999999/اماء شبكة.html` 显示。而浏览器设置是不显示 path 部分，所以将以空白显示，最终造成了 URL Spoof。这也是一个逻辑漏洞。多个 9999 把真正的域名滚动到了地址栏之外。

Safari show website address

show full website address

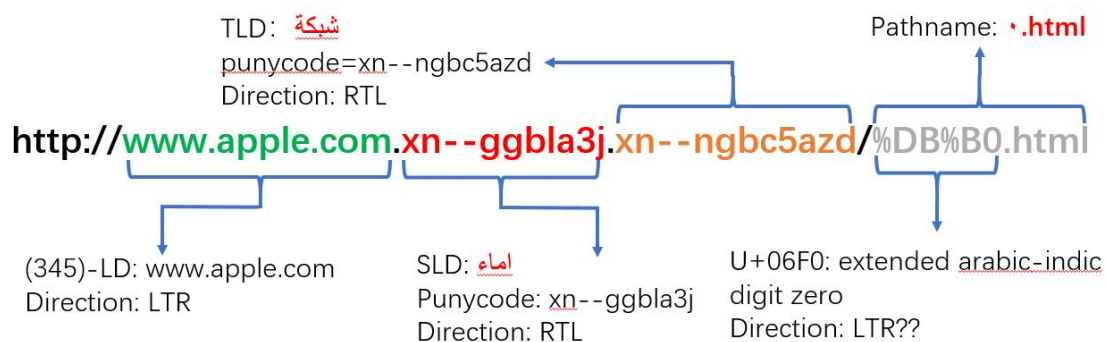


only show domain



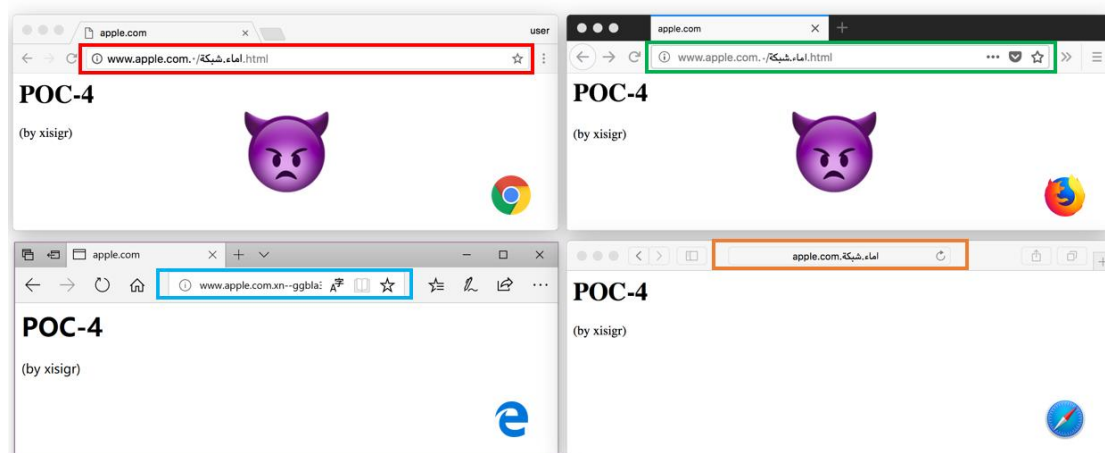
(5) POC-4

POC-4



可以看到，这个 POC-4 在 Chrome 和 Firefox 浏览器中的渲染欺骗性非常强。

POC-4



通过上面的几个案例分析，可以看到大多是和 pathname 有关，如果只显示 origin 是否可以解决这个问题？以 Safari 为代表，它就只显示域名，但在上面的例子中我们看到了，也会存在问题，pathname 被隐藏替换成空白符导致 Spoof 发生。那么，如果只是纯显示 origin 呢，把 pathname 在地址栏中彻底去掉？这也可能存在问题，并不能彻底解决 RTL Spoof 的问题。因为 origin 中也可以使用多个方向的字符，整体上产生乱序，造成用户从视觉上难以理解，从而产生欺骗。

五、 组合字符序列带来的安全风险

在字体排印学中组合字符（Combining character）是用来改变其它字符所用的字符。在拉丁文字中，最常见的组合字符为附加符号（包含重音号）。例如组合字符序列（Combining character sequence）：`kýõñ`（U + 006B U + 0301 U + 0075 U + 032D U + 006F U + 0304 U + 0301 U + 006E）。字符和组合字符序列是否相同，这取决于对于程序员来说，多数情况一个 Unicode 代码点代表一个单独的字符。但对于最终用户来说，它可能不是。对于最终用户认为的字符而言，更好的词是字形：在特定的书写系统中最小的

独特的书写单位。攻击者可以利用组合字符序列，绕过很多类似“黑名单”策略的防御。


如果组合字符序列的域名不被浏览器解析为 punycode 编码，就有可能产生视觉欺骗。

例如 google.com 域名中的字符+组合字符后，如果也不被使用 punycode 编码，视觉就可能发生欺骗。googlè.com 和 google.com，用户很难分清楚，并且在分辨率很高的显示器中，需要“鹰睛”去识别。但是，Chrome 显然是对例子中的组合字符做了 punycode 编码。很多情况下，IDN 域名都会被转换为 punycode 去解析。


CVE-2018-4260 是我之前发现的一个漏洞，利用了字符组合序列导致了 URL 地址栏欺骗。下面就以这个漏洞为例进行讲解。

在这个漏洞中，我们用到希伯来字符的 U+05D5 U+05B9 U+05E1 来完成漏洞攻击。


	059	05A	05B	05C	05D	05E	05F
0		א	ב	ג	ד	ה	ו
1	א	ב	ג	ד	ה	ו	ז
2	ח	ט	י	כ	ל	מ	נ
3	ס	ע	פ	צ	ק	ר	ש
4	ת	י	ך	ם	ן	ס	ז
5	ח	ט	י	כ	ל	מ	נ
6	א	ב	ג	ד	ה	ו	ז
7	ח	ט	י	כ	ל	מ	נ
8	א	ב	ג	ד	ה	ו	ז
9	ח	ט	י	כ	ל	מ	נ
A	א	ב	ג	ד	ה	ו	ז
B	ח	ט	י	כ	ל	מ	נ
C	א	ב	ג	ד	ה	ו	ז
D	ח	ט	י	כ	ל	מ	נ
E	א	ב	ג	ד	ה	ו	ז
F	ח	ט	י	כ	ל	מ	נ



hebrew letter vav (U+05D5)



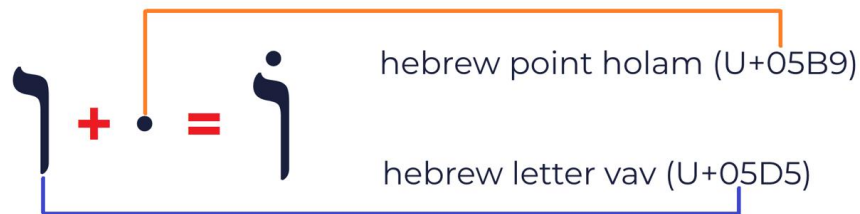
hebrew point holam (U+05B9)



hebrew letter samekh (U+05E1)

将字符 U+05D5 和组合字符 U+05B9 合成字符组合序列。

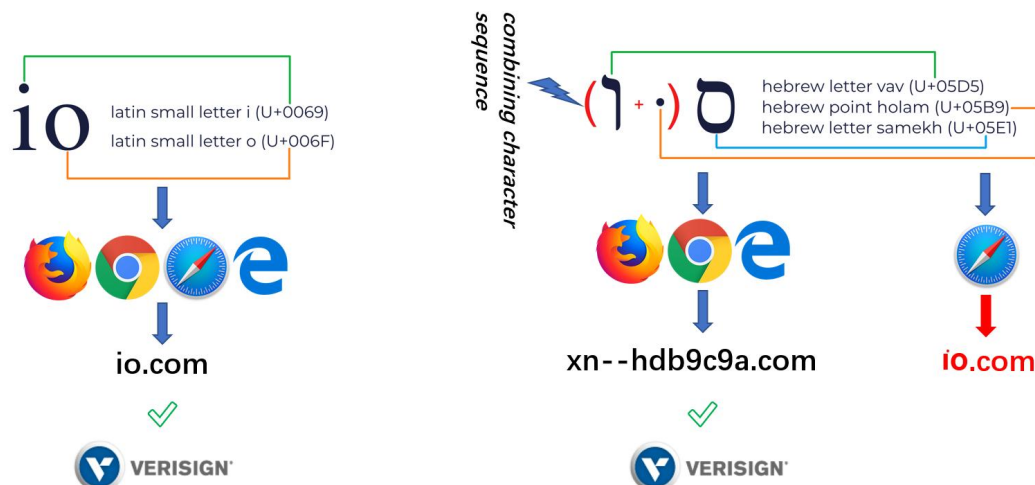
- Character + combining mark = *combining character sequence*



我注册了希伯来的这个域名，拉丁文的 io.com 和希伯来的 **יּו.כּוּם** 在浏览器中的显示如下。

希伯来的 **יּו.כּוּם** 域名在 Safari 浏览器中没有转换为 punycode。

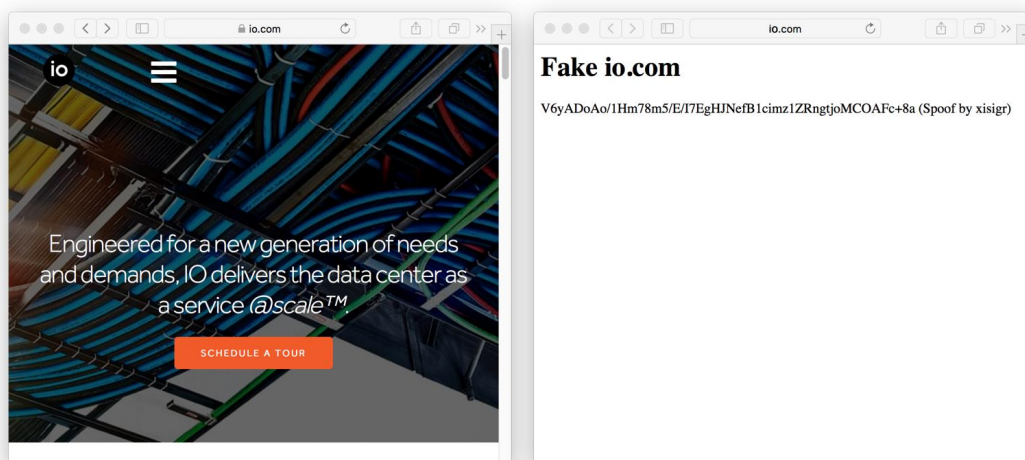
Latin: io.com VS Hebrew: יּו.כּוּם



最终的效果如下图，可以看下图中，左边是真实的 io.com，右边是 **יּו.כּוּם**。这两个域名在

Safari 中的视觉显示几乎一样。

Latin: io.com VS Hebrew: יו.קם



是否应支持组合字符序列进行注册？能否在注册阶段严格检查和防止此类恶意意图？

在 URL 中思考这些问题，你可能会发现一些新的东西。

六、 结语

Unicode 标准的出现和支持它的工具的使用是最近全球软件最重要的趋势之一，Unicode 为每个字符提供一个唯一编号，无论平台、程序或语言是什么，每个软件开发者都绕不开 Unicode 字符集。Unicode 目前分为 17 个平面，每个平面拥有 65536 个代码点，也就是说 Unicode 目前可以编码 $65536 \times 17 = 1114112$ 个字符。Unicode 1.0.0 从当年 1991 年发布的 24 个脚本 7161 个字符，发展到现在 2022 年 9 月 Unicode 15.0 的 161 个脚本 149,186 个字符。在这之后 Unicode 中一直会源源不断增加新的字符，软件处理这些字符时都可能因字形渲染不足、混合脚本、PunyCode、双向文本、组合字符等而导致漏洞触发，进而发起各式各样的网络攻击。在由安全漏洞核心驱动的网络攻击事件大量爆发的今天，对现有及尚未开发的 Unicode 编码、区域进行深入安全研究，从根技术出发去分析和控制漏洞收敛进而保护数据安全，是看似艰难却又是捷径的一条路。

七、 参考

1. Unicode 15.0.0

<https://unicode.org/versions/Unicode15.0.0/>

2. UNICODE SECURITY CONSIDERATIONS

<https://www.unicode.org/reports/tr36/>

3. UNICODE BIDIRECTIONAL ALGORITHM

<https://unicode.org/reports/tr9/>

4. UNICODE IDNA COMPATIBILITY PROCESSING

<https://unicode.org/reports/tr46/>

5. Characters and Combining Marks

http://unicode.org/faq/char_combmark.html

6. Homoglyph

<https://en.wikipedia.org/wiki/Homoglyph>

7. IDN Visual Security Deep Thinking

<https://xlab.tencent.com/en/uploads/2019/02/idn-visual-security-deep-thinking.pdf>

8. Domain Name Registration Process

<https://whois.icann.org/en/domain-name-registration-process>

9. INTERNATIONALIZED DOMAIN NAMES (IDNS) Registration Rules

https://www.verisign.com/en_US/channel-resources/domain-registry-products/idn/idn-policy/registration-rules/index.xhtml

10. Unicode Character Table

<https://unicode-table.com/en/1F946/>

11. UA720

<http://www.unicode.org/charts/PDF/UA720.pdf>

12. Bad Characters: Imperceptible NLP Attacks

<https://arxiv.org/abs/2106.09898>

13. 区块链黑暗森林自救手册

<https://darkhandbook.io/>