PunyVis: A Visual Analytics Approach for Identifying Homograph Phishing Attacks

Brett Fouss[*] Dennis M. Ross[†] Allan B. Wollaber[‡] Steven R. Gomez[§]

MIT Lincoln Laboratory

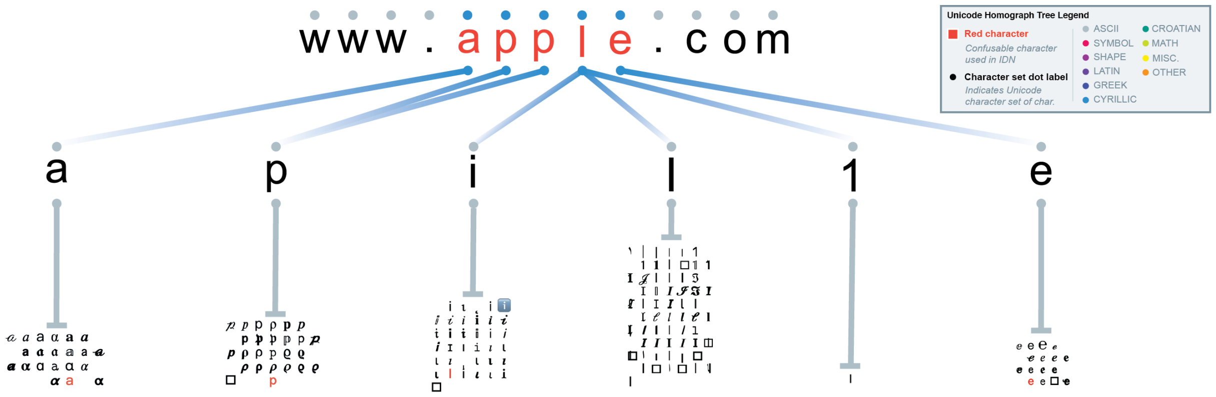

w w w . a p p l e . c o m
a
p
i
l
1
e
Unicode Homograph Tree Legend
Red character
Confusable character used in IDN
Character set dot label
Indicates Unicode character set of char.
ASCII
SYMBOL
SHAPE
LATIN
GREEK
CYRILLIC
CROATIAN
MATH
MISC.
OTHER

Figure 1: A Unicode homograph tree visualization in the PunyVis tool that explains how an internationalized domain name uses Cyrillic characters to spoof the domain name "www.apple.com" by Apple Inc. The top row shows the original domain name with Cyrillic characters highlighted in red, and each mapped to visually similar ASCII characters in the middle row. The bottom row shows other visually similar Unicode characters.

Abstract

Attackers seeking to deceive web users into visiting malicious websites can exploit limitations of the tools intended to help browsers translate domain names containing non-ASCII characters, or internationalized domain names (IDNs). These attacks, called homograph phishing, involve registering Unicode domain names that are visually similar to legitimate ones but direct users to distinct servers. Tools exist to identify when domains use non-ASCII characters, which get translated by the Punycode protocol to work with the Domain Name System (DNS); however, these tools cannot automatically distinguish between benign use cases and ones with malicious intent, leading to high rates of false-positive alerts and increasing the workload of analysts looking for evidence of homograph phishing.

To address this problem, we present PunyVis, a visual analytics system for exploring and identifying potential homograph attacks on large network datasets. By targeting instances of Punycode that use easily-confusable ASCII characters to spoof popular websites, PunyVis quickly condenses large datasets into a small number of potentially malicious records. Using the interactive tool, analysts can evaluate potential phishing instances and view supporting information from multiple data sources, as well as gain insight about overall risk and threat regarding homograph attacks. We demonstrate how PunyVis supports analysts in a case study with domain experts, and identified divergent analysis strategies and the need for interactions that support how analysts begin exploration and pivot around hypotheses. Finally, we discuss design implications and opportunities for cyber visual analytics.

[*]email: brett.fouss@ll.mit.edu
[†]email: dennis.ross@ll.mit.edu
[‡]email: allan.wollaber@ll.mit.edu
[§]email: steven.gomez@ll.mit.edu

Keywords: visual analytics, visualization design, cyber security, human factors, homograph phishing, Unicode.

Index Terms: Human-centered computing—Visualization—Visualization application domains—Visual analytics; Security and privacy—Systems security—Browser security

1 Introduction

The Domain Name System (DNS) is a ubiquitous system that enables Internet and web services to be more usable by translating human-readable domain names into the numerical Internet protocol (IP) addresses used by computer networks. However, cyber attacks have targeted this translation process in order to deceive users into accessing unintended—and potentially malicious—servers.

Homograph phishing attacks focus on deceiving Internet users by crafting domain names that are visually-similar to other domain names and therefore are *confusable* by the user. A user viewing a displayed confusable name (e.g., a web Uniform Resource Locator, or URL) then might click a hyperlink to this URL in their web browser expecting to navigate to one site, only to be redirected to a different site without their knowledge.

The homograph phishing problem presents two key challenges for cyber visualization practitioners and researchers. First, the problem is rooted in the visual confusability of domain names, which is compounded by the need to render Unicode characters for diverse user groups and human languages. For example, two visually-similar internationalized domain names (IDNs) are easily distinguishable by a computer because it operates on domain name characters' underly-

DISTRIBUTION STATEMENT A. Approved for public release. Distribution is unlimited. This material is based upon work supported under Air Force Contract No. FA8702-15-D-0001. Any opinions, findings, conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the U.S. Air Force.

ing representations rather than character glyph renderings. However, humans browsing the web generally do not see the underlying encodings; in fact, hiding low-level address representations (i.e., IP addresses) is the purpose of DNS. Unicode characters that are rendered similarly, or in some cases identically, are able to deceive people who might also lack the knowledge or attention to avoid falling prey to the attack [7].

At the same time, forensic tools and processes for analyzing DNS records generally do not provide all the context needed to identify evidence of homograph phishing in logs. IP addresses in logs are not obviously malicious at a glance, nor is a mapping between an address and a potentially confusable domain, because addresses of legitimate servers can often change over time. This creates a need for tools that let humans identify past instances of potential phishing, incorporating information about how Unicode and the translation protocol, called Punycode, can be exploited by attackers.

We present PunyVis, a visual analytics system that supports analysts in assessing the threat of Punycode URLs recorded in DNS logs. The system is designed to support cyber analysts looking at DNS records gathered over time and assessing the security risk of queries from hosts within an enterprise network. It provides both a list of records from the logs to explore, ranked according to a threat-assessment algorithm we created, as well as an exploratory user interface.

The PunyVis system is uniquely tailored for forensic investigations in which a determination must be made as to whether users in an enterprise network fell prey to homograph attacks using Punycode. The visual interface includes a novel visualization we call a *Unicode homograph tree*, which uses tree diagrams, color encoded glyphs, and multiple typeface renderings to explain how a domain might be formed with intent to spoof legitimate domains. This tool assists analysts in spotting domains with a deceptive visual appearance that may have been accessed unwittingly by users in the network. In addition, we contribute a dashboard layout and query interface for exploring these logs that includes a Punycode-enabled search bar, as well as one-click filters for DNS-related information. Through a case study of analysts using the system, we found evidence that the interactive features and visual representations helped analysts more effectively identify and explore instances of these attacks.

We make the following research contributions:

- The design and implementation of a novel visual analytics system targeted at identifying domain homograph phishing attacks—the first to combine an algorithm for triaging homograph risk with interactive visualization;
- An algorithm that ranks potential phishing attempts with Punycode-encoded domain names in DNS logs by deceptiveness;
- A novel visual representation of Punycode-encoded domain names called the Unicode homograph tree, which helps users understand potential visual confusion about individual domain name characters;
- Anecdotal design feedback from a case study with cyber security analysts showed that PunyVis was able to help their Punycode-exploration workflow and allow them to quickly and efficiently triage DNS records for Punycode phishing attempts.

The remainder of this paper is organized as follows: in Section 2 we discuss the homograph attack problem and the systems it targets in greater depth; in Section 3 we present an algorithm for ranking potential malicious records based on homograph confusability, along with a novel interactive interface that allows an analyst to efficiently explore these records; in Sections 4 and 5 we discuss a case study with subject-matter experts and their feedback, and identify open challenges; finally, in Section 6 we conclude and summarize our key findings.

Table 1: Example of visually similar Unicode homographs of 'E'.

Visual Representation	Unicode Encoding	Alphabet
E	U+0045	Latin
Е	U+0415	Cyrillic
Ε	U+0395	Greek and Coptic
𖼭	U+16F2D	Miao
𝖤	U+1D5A4	Mathematical SS

2 Background and Related Work

2.1 Domain Name Translation and Punycode

The DNS service was created to make addressing Internet servers easier and less error-prone for humans. It is ubiquitous, helping to translate usable, memorable strings of characters (domain names) into numerical IP addresses. In general, DNS servers maintain mappings between domain names and IP addresses, and client DNS-lookup results are typically requested and cached by a device's web browser or operating system.

DNS was initially designed to interpret only ASCII characters for domain names. This implementation choice prevented the use of non-Latin standard characters that are necessary to represent additional languages. To overcome this limitation, IDNs were developed to include an expanded character set specified by Unicode [18]. This standard currently includes encodings for over 100,000 characters and symbols.

Unicode-to-ASCII translation using Punycode. To avoid re-architecting the DNS system to include the expanded Unicode-containing IDNs, a translation protocol called Punycode was developed and published in RFC 3492 [4] as part of a standard for Internationalizing Domain Names in Applications (IDNA) ([8], updated in [5]). Punycode and IDNA are used by applications to deterministically translate portions of IDNs containing Unicode (called "U-labels") into ASCII-compatible encoding (ACE) labels ("A-labels") that result in domain names usable by DNS services.

An example of this translation process for a mixed ASCII and Unicode ".com" IDN is shown in Figure 2. First, all ASCII characters are copied into the output A-label in order. If any were copied in this way (i.e., not all characters were Unicode), an ASCII hyphen ('-') is then appended to the label. Next, each non-ASCII character is encoded based on its position and character type, or code point. A finite state machine determines and appends these character encodings to the end of the new label. The top-level domain (TLD) is then appended if it is all ASCII, otherwise the process is repeated for the TLD. Finally, the ACE prefix `xn--` is prepended on the A-label to differentiate Punycode-encoded ASCII domains from native ASCII domains.

Homographs and character confusion. As a result of the increased character set for IDNs and the similarity between many language alphabets, some domain names appear visually similar or indistinguishable to one another despite containing distinct characters. These visual similarities are caused by *homoglyphs*. Homoglyphs are characters that are visually similar to one other. The Unicode Foundation identifies over 6,000 characters and character sets that are considered homoglyphs even though their underlying representation in Unicode is distinct. Distinct domain names in Unicode that contain homoglyphs are *homographs* if the domain names are visually similar. Because homoglyphs can also be homographs, we will use the term homograph throughout this paper unless the distinction needs to be explicit.

Although Punycode allows for domain names in most native languages to be supported by DNS, it unintentionally introduced some security concerns for end users of IDNA applications, like web

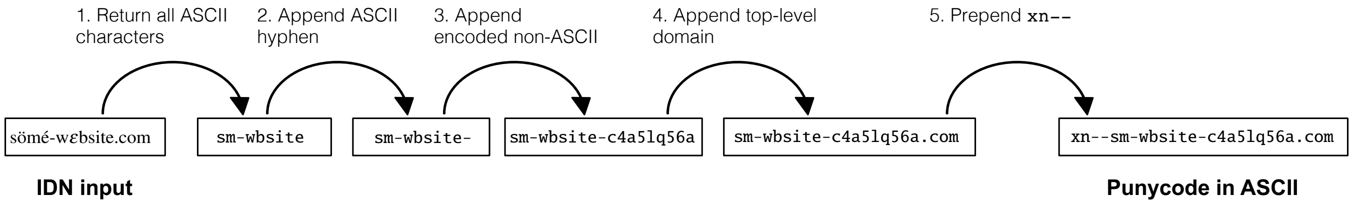


Figure 2: Steps for translating an IDN with an ASCII top-level domain to an ASCII compatible encoding. See RFC 3492 [4] for details about encoding the non-ASCII characters with Punycode (step 3).

browsers. For the character 'E', Table 1 shows a non-exhaustive sample of confusable characters, including visually-indistinguishable homoglyphs (in Latin, Cyrillic, and Greek and Coptic character sets) as well as visually-similar ones (Miao and Mathematical Sans-Serif), along with their distinct Unicode encodings. One key concern revolves around applications presenting visual renderings of these characters to users while trusting them to distinguish between the underlying character types. While some issues are handled by a validation process required by the IDNA standard called NamePrep [13], for example preventing domain names with generally-invisible characters, other opportunities for registering domain homographs remain, as discussed in the following section. Next, we discuss how attackers capitalize on human confusion involving homographs in phishing attacks.

2.2 Homograph Phishing Attacks and Cyber Operations

Phishing attacks are a family of cyber attacks in which attackers mimic trusted users, emails, websites, or other communications in an attempt to extract information or distribute malware to their victims. The existence of homographs in Unicode and the requirement for Punycode to be used with IDNs for DNS translation leads to potential phishing attacks. A computer is able to distinguish domain homographs, but because they are visually similar—or worse, identical—a human may interpret them as being the same domain. When many web browsers encounter URLs with non-ASCII characters from a single language in Unicode, they simply render the text rather than display the Punycode encoding. Because of this, malicious actors are able to craft and register domains that are visually identical to known or trusted websites. These visually-indistinct domains would then be mapped by DNS to IP addresses controlled by the malicious actor, who provides the deceptive URL to victims in the form of email or web links, etc.

Web browsers do not always show when the Punycode protocol is being used, allowing this attack to propagate widely. Users may click on a non-ASCII link mimicking a legitimate website and be routed to a malicious impostor without being able to visually distinguish the difference in the URL. This security vulnerability was first disclosed by Gabrilovich and Gontmakher in 2002 [11]. More recently, in 2016, a security researcher created a proof-of-concept by registering the web domain "apple.com" (`xn--80ak6aa92e.com`), which remains online as of August 2019, using only non-ASCII characters to mimic the domain of the legitimate site from Apple Inc. [22]. Liu et al. recently provided a review of IDN abuses and demonstrated that homograph attacks remain a problem [15]. They used a similarity-based image comparison between renderings of the Alexa Top 1K single-level domains and 1.4 million IDNs, finding over 1500 registered homographic IDNs, with fewer than 5% of them being registered for brand protection. Furthermore, they were able to successfully register 10 homographic IDNs with GoDaddy.

For enterprise network security, analysts interested in identifying examples of successful homograph attack campaigns face issues of scale. Although simple regular expressions can filter all Punycode URLs by a search for `xn--`, major deficiencies exist in using this approach to identify potential attacks. First, enterprise networks typically have millions of DNS requests daily. Large numbers of these requests may have been encoded with Punycode. Second, Punycode is required for IDN lookups, and we assume the majority of IDNs are legitimate—especially for servers located in regions where people regularly use non-Latin character sets. Taken together, these factors cause the process of determining which Punycode domains indicate risk to be largely manual and slow. Because the Punycode encoding of an IDN is not intuitive at a glance, instances of Punycode must be translated back to Unicode for human readability. Third, to the best of our knowledge there exist no tools that let analysts quickly synthesize supporting information about these domains.

2.3 Interface Design for Homograph Attack Protection

Homograph attacks, and most other kinds of phishing, are cyber attacks that exploit human perception and human attention [7]. Therefore, countermeasures against such attacks should aim to provide contextual information to Internet users to decrease their risk of unknowingly engaging with phishing websites.

Previous work on protecting users from malicious homograph attacks has involved modifications to web browser interfaces to expose the underlying Unicode encoding of a URL—in particular, highlighting characters based on each one's character set [20] or using alerting users when IDNs or mixed character sets are used [14].

A more direct countermeasure is to unconditionally display URL ASCII representations instead of Unicode IDNs in the browser address bar; however, this might unfairly affect people who regularly use languages with non-Latin character sets, and in fact runs contrary to the premise of IDNs in the first place. Nevertheless, to combat homograph attacks, web browser vendors have adopted measures to show ASCII representations of URLs in certain cases. Firefox and Chrome, developed by Mozilla and Google respectively, combine whitelisting in addition to simple rule-based policies to make this determination [12, 16]. Though this provides a layer of protection in the browser, it does not prevent users from clicking on deceptive links attained through other platforms, such as in the body of an email. These defensive approaches are application specific.

Browser defenses and other application defenses nudge users toward safer behaviors, but they cannot prevent instances of successful homograph phishing outright. Therefore, analysts looking for forensic evidence of threats need tools that support finding, analyzing, and assessing the risk of such attacks in network logs. Given that poor visual perceptual of characters is the underlying vulnerability in these types of attacks, tools addressing the problem likely need to present visual evidence to security stakeholders to help explain any risk assessments. Hence, we believe a visual analytics system with a human in the loop is suited to this problem. Several factors make it difficult to decide automatically whether a domain name is ultimately a phishing trap, including the geolocations of the DNS-queried servers, language considerations, and perceptual factors affecting glyph legibility. Furthermore, analysts might follow up possible threats by visiting malicious domains in order to determine the impact of any unwitting downloads by users; as such, even automated systems that perform human-like perception and reading—for example, using machine learning techniques like generative adversarial networks (GANs) to break CAPTCHA systems [19, 21]—may not have enough general knowledge about cyber security or data science to complete the threat assessment without supervision.

In designing a visual analytics system for this problem, we extended interface design choices and rule-based algorithms from prior research. We discuss the algorithm and visual design of PunyVis in the following section.

3 PunyVis Design

The design of PunyVis is aimed at supporting typical threat-hunting practices for large logs, where an analyst would start by creating and applying a series of carefully selected filters to search for abnormal, potentially malicious events in their logs. PunyVis includes an *algorithm* that helps analysts rank and triage records during exploration, along with a *visual interface* that supports the analyst in viewing related information about records and interacting with additional filter controls.

3.1 Algorithm

In order to prioritize domain names that are most likely to indicate Punycode-based phishing attempts, we designed a simple algorithm, shown in Figure 3a, that can effectively triage high-volume DNS records. PunyVis is able to extract and process many common types of datasets containing DNS records, including packet capture (PCAP) and processed PCAP from software like Zeek (formerly Bro) [3].

This algorithm was designed to recognize and prioritize Punycode-encoded DNS requests that are possibly malicious and targeted at deceiving English speakers. The built-in filtering tools could be adjusted to aid in Punycode investigations targeting additional languages as well. The algorithm's filtering steps provide a ranking of the available DNS records by determining the risk posed by a particular URL. Each of these filters was chosen to match the standard data science processes an analyst exploring Punycode would likely use.

3.1.1 Algorithm Intuition

The algorithm uses a sequence of four rules to determine the risk score of a domain name. All Punycode-encoded domain names are prepended with `xn--`, so the first filtering step can simply split the data into examples that begin with `xn--` and those that do not. The algorithm then determines the risk posed by the Punycode-encoded DNS records. The second filtering step scans the entire URL and then creates a set of phishing-candidate URLs. These candidates are found by replacing all possible Unicode substitutions for any character that is not from the character set used in standard English (the ASCII character set). Next, the candidate domain names are screened. A check is done to determine if any of them is entirely representable using only ASCII characters, implying that the URL could have been rendered as a Punycode-free homograph. Finally, any candidates that match ASCII-representable words or phrases are then filtered as to whether or not they belong to the Alexa Top Million websites [2]. Any Alexa Top Million websites that contain Punycode were removed prior to this check.

Each step of this algorithm checks for additional indicators that an IDN is being used as a homograph attack. The first filter simply finds if a site is using the Punycode protocol. It is impossible to have a Punycode attack without using the encoding, but there are also many websites that legitimately use Punycode. By itself, the first filter contains the range of Punycode-encoded domain names from benign to deceptive. Prioritization begins in the next step by identifying domain names that are somewhat or fully similar to potential English labels (entirely representable in ASCII). This is an attempt to separate IDNs that are legitimately encoding non-English domain names from ones that mimic English-language domain names using non-Latin characters. Finally, we compare the candidate domains to the Alexa Top Million website URLs, which can be periodically updated. If there is a domain that contains Punycode and is mimicking a popular website, it is likely the site was created by bad actors as a phishing website (or by security researchers exploring the Punycode exploit).

The Alexa sites can also be replaced by a list of enterprise-specific domain names that let an analyst identify phishing attempts on domains within their own network (i.e., from insider threats) or ones popular with the networks' users (i.e., that may be used in spear phishing) if they otherwise do not appear in the aggregated Alexa dataset.

Although the triage algorithm assigns a risk score of 1 through 4 to each domain name (see Figure 3b for interpretations of these scores), we chose to display a binary label to the analyst in the current implementation, providing simpler, more coarse-grained visual indicators for new users of the visual interface, which is discussed in Section 3. Therefore, all records are aggregated into one of two groups: records of likely benign events (categories 1 and 2) and records of potentially malicious events (categories 3 and 4). In the user interface, we represent each group with a green and red dot glyph respectively. Depending on the particular dataset, these preferences could be adjusted to balance the trade-off between a higher false-positive probability (broader risk categorization) and higher false-negative probability (more granular risk categorization).

3.1.2 Walkthrough for "evil.com"

As an example, we step through how the algorithm scores the domain name "evil.com" (`xn--evi-lhd.com`).

1. Initial state for input record. (**current risk score:** 1)

2. Determine the domain name `xn--evi-lhd.com` is Punycode by looking at the string prefix. (**current risk score:** 2)

3. Generate mapping of ASCII characters in the name to ones that are visually confusable in rendered Unicode. The characters 'e', 'v', and 'i' are all ASCII, but the 'l' is Cyrillic. It produces:

$$l \to \{1, I, l, i, L\}$$

4. Enumerate all names that swap in the confusable variants in place of their input characters. We consider these as *candidates* for what a user might have thought she was querying with DNS. In this case, it swaps in each variant of the Cyrillic 'l' to produce the set:

$$\text{evi1}, \text{eviI}, \text{evil}, \text{evii}, \text{eviL}$$

5. Determine whether any of the candidates is ASCII-representable. Here, "evil" is entirely represented in ASCII, implying that a homograph exists that does not require Punycode for rendering. (**current risk score:** 3)

6. Query Alexa Top Million websites list for each of the candidate URLs.

$$\text{evi1.com}, \text{eviI.com}, \text{evil.com}, \text{evii.com}, \text{eviL.com}$$

   Because none of these is on the list, the score remains the same. (**current risk score:** 3)

7. Return score. The final risk score is 3, with our qualitative interpretation as *possible Punycode phishing*.

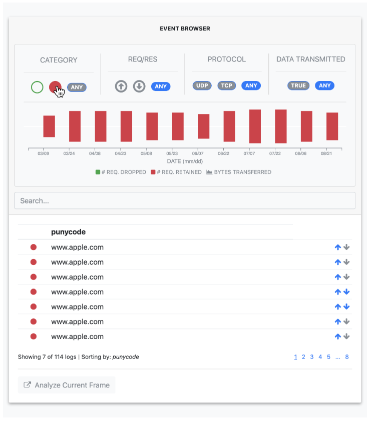
```
Input domain from DNS query log
$risk \leftarrow 1$
if domain is Punycode encoding then
    $risk \leftarrow 2$
    Check if any characters in domain are confusable
    $D \leftarrow$ all confusable variants of the domain name
    $isAllASCII \leftarrow$ True iff any variant in $D$ is all ASCII
    $isAlexa \leftarrow$ True iff any variant in $D$ on Alexa list
    if $isAllASCII$ then
        $risk \leftarrow risk + 1$
    end if
    if $isAlexa$ then
        $risk \leftarrow risk + 1$
    end if
end if
return $risk$
```

(a) Algorithm to calculate Punycode homograph phishing risk.

Determination	Risk Score	Qualitative Risk
No Punycode encoding in URL	1	No Punycode phishing
Candidate words or phrases not ASCII-representable	2	Unlikely Punycode phishing
Candidate words or phrases are all ASCII-representable	3	Possible Punycode phishing
Matches Alexa Top Million or site-specific list	4	Likely Punycode phishing

(b) Summary of assessments for Punycode domain names.

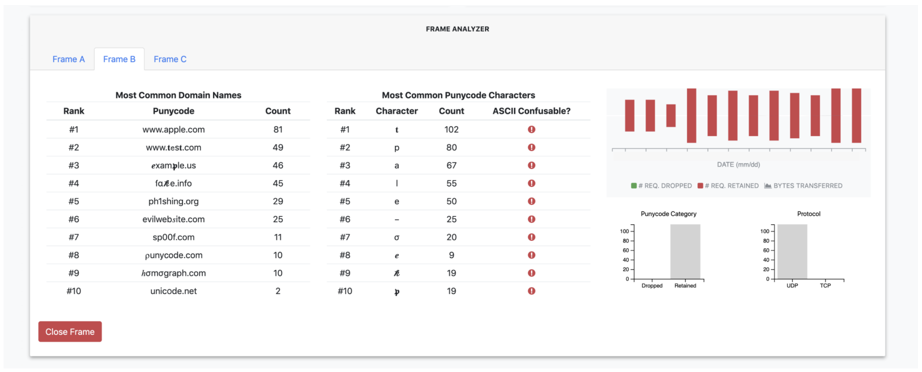
Figure 3: Scoring Punycode domains based on suspicious of phishing.



	punycode
●	www.apple.com
●	www.apple.com
●	www.apple.com
●	www.apple.com
●	www.apple.com
●	www.apple.com
●	www.apple.com

Showing 7 of 114 logs | Sorting by: *punycode* 1 2 3 4 5 ... 8

(a) Event browser showing filters, timeline, and search bar.



Most Common Domain Names

Rank	Punycode	Count
#1	www.apple.com	81
#2	www.test.com	49
#3	example.us	46
#4	faƙe.info	45
#5	ph1shing.org	29
#6	evilwebsite.com	25
#7	sp00f.com	11
#8	ρunycode.com	10
#9	hσmσgraph.com	10
#10	unicode.net	2

Most Common Punycode Characters

Rank	Character	Count	ASCII Confusable?
#1	t	102	●
#2	p	80	●
#3	a	67	●
#4	l	55	●
#5	e	50	●
#6	–	25	●
#7	σ	20	●
#8	e	9	●
#9	ƙ	19	●
#10	þ	19	●

(b) Frame analyzer view showing summary statistics of user-filtered data frames.

Figure 4: User interface components supporting searching, filtering, browsing, and aggregated data analysis in PunyVis.

3.2 Interface Design

This section describes the overall layout of the PunyVis interface and the design of each of its components.

The user interface includes a query interface tailored towards searching through network data containing Punycode-encoded domains. It additionally supports visualizations and displaying log information to help analysts evaluate potential risk and pivot to other tools. Summary visualizations give an overview of Punycode and DNS-related information to provide understanding of homograph attack potential across an entire network dataset.

The PunyVis dashboard consists of three main components: (1) the event browser, (2) the log inspector, and (3) the frame analyzer. Here, we discuss the design and function of each.

3.2.1 Event Browser

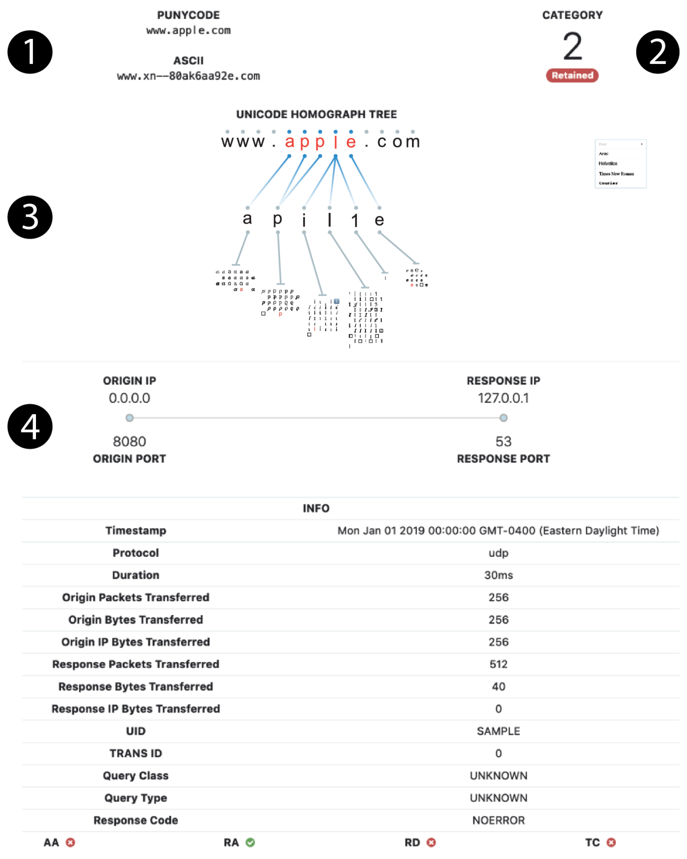
The *event browser* view, shown in Figure 4a, consists of a table showing all available log entries, as well as helpful filtering utilities designed to help users find potentially-malicious events quickly, while permitting exploration of all DNS records containing Punycode IDNs.

The table view shows a list of all DNS records that contain Punycode. Non-Punycode records (DNS records that are not prefixed with the string `xn--`) are automatically filtered out[1]. In the table, the domains are displayed using their Unicode representations for two reasons. First, because their Punycode representations are not human interpretable, and second, because it is helpful for analysts to see the what potential phishing targets would see in their browser search bar. Each row in the table also contains a dot glyph indicating whether the domain has been flagged as potentially malicious or not, as well as two arrow icons indicating whether (1) a request was successfully made to the domain name server, and (2) whether the domain name server was able to resolve the domain name and complete the query.

Although exploratory analysis is encouraged, we expect most analysts using the tool first to seek answers to a few common questions. Because of this, PunyVis contains a quick-filter panel allowing the user to filter the records based on the following parameters:

- Was the domain flagged as potentially deceptive?

[1]Simply filtering for the `xn--` prefix does not handle cases using only ASCII characters (e.g., an uppercase *I* character substituted with a lowercase *l* character), which may be visually confusable within certain type faces and for which the `xn--` prefix is not encoded. The back-end analytic can optionally be set to handle such cases.



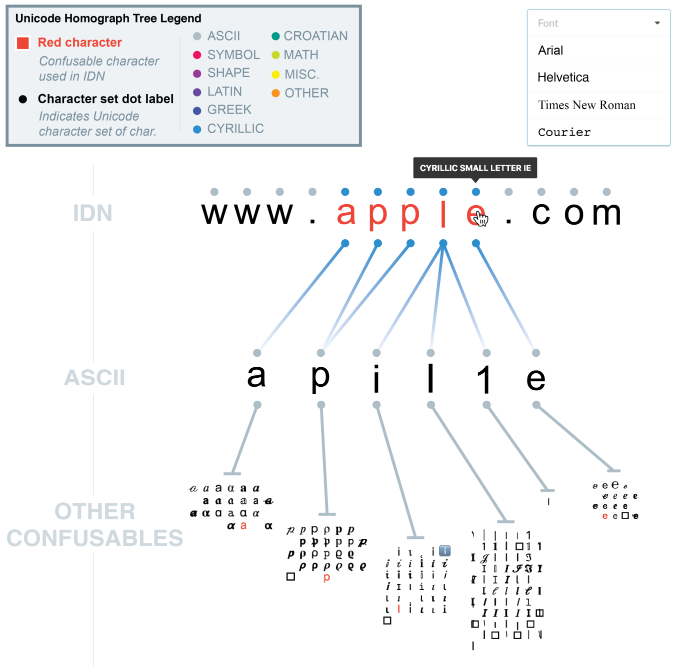
ORIGIN IP 0.0.0.0 — RESPONSE IP 127.0.0.1

8080 ORIGIN PORT — 53 RESPONSE PORT

INFO	
Timestamp	Mon Jan 01 2019 00:00:00 GMT-0400 (Eastern Daylight Time)
Protocol	udp
Duration	30ms
Origin Packets Transferred	256
Origin Bytes Transferred	256
Origin IP Bytes Transferred	256
Response Packets Transferred	512
Response Bytes Transferred	40
Response IP Bytes Transferred	0
UID	SAMPLE
TRANS ID	0
Query Class	UNKNOWN
Query Type	UNKNOWN
Response Code	NOERROR

AA ✗ RA ✓ RD ✗ TC ✗

(a) Overview of the PunyVis log inspector view.



(b) Detailed view of the Unicode homograph tree.

Figure 5: The log inspector view showing network metadata extracted with the Bro tool. In (a), the overview shows the main dashboard components, including ❶ the domain name rendering and encoding, ❷ the threat categorization label determined by the risk algorithm, ❸ the Unicode homograph tree, and ❹ query metadata details. In (b), the Unicode homograph tree illustrates how visually-indistinguishable homographs from the Cyrillic character set are used to spoof a common domain name.

- Was a request successfully received by the DNS server, and/or was the request successfully resolved?
- What protocol did the transaction use?
- Was data transmitted?

These filters aim to streamline the analysis process by allowing an analyst to quickly examine the highest-risk events, and then spend any remaining time examining progressively lower risk events.

In most situations, the highest-risk records would be those that:

(a) contain a domain with ASCII-confusable Punycode, and

(b) are resolvable by a DNS server, and

(c) exchanged data.

If (a) and (b) are both true, then the record strongly indicates that the domain name was formed with malicious intent to spoof. If both are false, the risk is very low. If (a) is true but not (b), it is likely that a user incorrectly typed a domain name that contained confusable characters. This may be accidentally caused by the use of a non-U.S. style keyboard; in this case, the risk is lower because a server matching the domain name was not found. If (a) is false and (b) is true, the associated web site is likely legitimate; the use of IDNs is normal and usually the presence of non-ASCII characters in an IDN is visually detectable by users. In suspicious cases, (c) can help indicate whether or not the phishing attempt was successful—that is, whether the user actually sent information to the spoofed server. In this, case additional investigation is required to determine the effect (e.g., information was lost, malware was delivered).

A timeline visualization shows the ratio of dropped-to-flagged DNS records containing Punycode over time, while also providing a scrubber to filter the table by time. A relatively high number of temporally clustered, flagged events might indicate an orchestrated phishing campaign, like routinely-performed penetration testing via phishing frameworks. Such a case would be visually represented in this graph by a larger-than-average red bar. The analyst could use the timeline scrubber to quickly isolate the time range of this event and continue their investigation accordingly.

When searching through DNS logs containing Punycode-encoded strings using a standard search bar, the underlying complexity of Punycode encodings and particular Unicode code points must be abstracted away from the user. Furthermore, simple string matching would not equate ASCII letters with similar letters from other Unicode character sets, such as the Cyrillic character set. Therefore, PunyVis offers a smart search bar that matches all permutations of confusable domains for a single ASCII query. Analysts can then confidently search for domain names using a standard keyboard without having to know exactly how those domain names might be spoofed in the dataset. This could also be extended to ideogram languages by allowing additional input encodings for the search feature and matching these inputs against known homographs in a language- or site-specific list.

3.2.2 Log Inspector and Unicode Homograph Tree

In defensive cyber operations involving network security, it is often necessary for cyber analysts to drill down and view raw network log information. This task is meant to provide concrete evidence of supporting indicators of compromise to perform a comprehensive threat assessment, or to track down correlated data points between tools. To support these steps, PunyVis incorporates a details-on-demand *log inspector* view on the right-hand side of the dashboard.

For a selected record, the log inspector shows supporting network metadata collected from logs pertaining to that record. Figure 5a shows the data attributes included in the log inspector. Analysts are shown the selected domain name both in Unicode and Punycode encodings, as well as the result of the threat categorization algorithm.

By their design, homograph phishing attacks aim to exploit the visual perception of humans. To help analysts examine the appearance of IDNs within network logs and evaluate if each domain could be a part of a Punycode attack, we present a new of visualization of domain name records called the *Unicode homograph tree*, as shown in Figure 5b. For domain names containing non-ASCII characters, the Unicode homograph tree offers an in-depth view of how spoofed domain names can be created using confusable Unicode characters. This allows analysts to better evaluate if a domain name was formed with the intent to spoof.

The tree visualization is composed of three distinct levels. The top level shows the original Unicode domain, with each confusable character highlighted in red. The middle level shows the set of all ASCII characters mapped to the red confusable characters in the original domain. The bottom level shows all possible confusable characters for each ASCII character in the middle level, with the particular confusable character used from the top-level highlighted again in red.

Our design builds on work done by Krammer [14] and Liu et al. [20] to visually encode homograph obfuscation by coloring Unicode strings according to character sets. We enrich the visualization with the tree view and colored dot indicators for each character to help users distinguish between Unicode character sets used in the domain. The user may also hover on a letter to read the description of the Unicode character.

Finally, a font selector dropdown offers the capability to view the appearance of the domain name in various serif, sans serif and fixed-width fonts. The font and typeface can make certain homographs less visually confusable or highlight when homograph attacks may have occurred. Using the Unicode homograph tree, analysts can perform an efficient and holistic evaluation of each flagged domain name by considering the character sets used, ASCII character confusability, and visual appearance across multiple typefaces.

By considering information from both the log inspector and the homograph tree, we expect to enable the analyst to assess the risk in a DNS log dataset more quickly and accurately than is currently possible.

3.2.3 Frame Analyzer

Finally, PunyVis offers overview and summary information in the *frame analyzer* view, shown in Figure 4b. This shows the most common domains, most common Unicode characters, most common Unicode character sets, a breakdown of protocols used, as well as trends of flagged events across DNS network logs.

In order to support the investigation of multiple hypotheses during forensic analysis, there is a need for cyber analysts using visual analytics tools to be able to queue several filtered data subsets to analyze at a later time. The frame analyzer is aimed at providing this functionality. As the analyst explores the dataset, intermediate states may be saved in the frame analyzer to be investigated later. High-level metrics about instances of Punycode and usage of Unicode characters, which are also visualized, might help network administrators design more secure policies or firewall configurations.

4 Case Study

We qualitatively evaluated PunyVis in a case study with two subject-matter expert (SME) analysts in order to gather design feedback and understand its strengths and limitations. In the study, the analysts explored how the tool supports their goal of finding indicators of compromise (IOCs) corresponding to potential homograph attacks in a dataset of DNS server query logs. Both were mid-career analysts supporting a large organization, and both had expertise in operational cyber threat analysis and using data science tools to support these operations. Neither had experience hunting for domain homograph attacks specifically. However, each analyst was given an introductory explanation by one of the authors about Unicode and the Punycode protocol, IDNs, and homograph phishing attacks.

4.1 Data and Tools

In speaking with the analysts, they noted that using a tabular view of the log data with filtering functions, for example in Microsoft Excel, would be a reasonable starting approach for exploring homograph IOCs. Based on this, we asked each analyst to show us and explain how they might use Excel for the task. Independently, each analyst used Excel to explore a dataset of 50,000 DNS records extracted into a comma-separated values (CSV) file using `Bro`. The data consisted of DNS records with Punycode-encoded domain names, enriched with connection metadata (such as transferred packet and byte counts). The analysts were not given specific instructions on how to perform the analysis and each spent about 30 minutes exploring the data.

After exploring and discussing how Excel supported their task, each analyst was given an overview of PunyVis and its features. The Alexa Top Million data used by the PunyVis algorithm during the trial had been downloaded on 27 April 2017. Each openly explored the same dataset using the tool and provided verbal feedback as they went. Roughly the same amount of time was spent by each analyst during this process as they spent with Excel.

4.2 Observations

Providing context for a starting point is helpful. Both analysts noted the need for a starting point when using Excel to look for homograph attack IOCs. Because they were not using a data analysis tool targeted at Punycode attacks, it was difficult to determine which domains (represented as strings like `xn--sm-wbsite-c4a5lq56a.com`, for example) contained obviously confusable or deceptive characters. After browsing around the spreadsheet, P_1 commented that "pure logs don't offer a starting point to investigate homograph attacks," adding that "there are too many valid Punycode domains that clearly aren't for phishing." Because the exported data file did not contain a header row, P_2 began by adding a row with column labels based on his inferences about the type of data it included. They also tried to sort rows by their timestamps, which is a typical starting approach for cyber forensics in attack campaigns where causal relationships exist within a time series.

By comparison, the ranked records and set of filters displayed by PunyVis seemed to provide a launching point for the analysts. Both analysts spotted "www.apple.com" in the visualization of suspicious domains early, then performed filtering or investigations of the associated IP addresses to discover that some instances contained confusable characters, evidence of possible phishing attempts.

Views that provide supplemental application-specific information enable repeatable cross-referencing. When exploring the data with both tools, P_2 identified cases where additional information could be quickly pulled in to better support their hypotheses. With Excel, they noted that if IP addresses could be quickly correlated with geographic locations, it would be easier to assess whether the use of an unexpected character set is suspicious. During the PunyVis session, they suggested an additional view that would incorporate `whois` domain registrar records when inspecting domain names.

P_1 noted that being able to save and revisit partial analyses in PunyVis' frame analyzer "streamlines analysis". They gave positive feedback about the Unicode character visualization ("I like the [Unicode homograph tree] visualization") and referred to record types by their visual attributes (i.e., exploring "the green ones", or dropped records), suggesting that the visual encodings impacted their own model of the data and how they planned to interact with it.

Analysts can have divergent strategies and needs for the same task. In general, we observed that P_1 and P_2 tended to favor top-down and bottom-up analysis styles, respectively. P_1 began by clicking on several of the quick filters and saving a few data frames to the frame analyzer for later analysis. Based on this broad exploration, P_1 decided on a few top-level domains (TLDs) (e.g., `.net`) to investigate. When a query returned results, P_1 could visually verify that none of the requests to domains with that TLD were resolved by the server. Thus, P_1 disregarded these domains and repeated the analysis with a new TLD.

In contrast, P_2 took more of a bottom-up approach, beginning by looking for individual records that met their criteria for being suspicious. They used the quick filters to look for records of domains that both contained confusable Unicode characters and where data transfer succeeded to and from the DNS server. As an example of this per-record investigation, P_2 was confused as to why a domain that looked like "www.apple.com" was marked as potentially malicious, then inspected the Unicode homograph tree and found it contained Cyrillic characters. P_2 noted that this would warrant further investigation.

Another individual difference we observed between analysts was how much access they wanted to the underlying algorithms and analytics feeding the visual interface. This was apparent when the analysts were using PunyVis but not Excel. It was noted that the latter did not include any behind-the-scenes processing of the data by the application—all analysis steps were performed manually. With PunyVis, P_1 was disappointed that its ranking algorithm was concealed behind the interface without a way to "get at" the analytic, but did not have specific concerns about how the algorithm worked. P_2 told us they trusted the algorithm when asked.

5 Discussion

In this section, we discuss the main lessons learned through our design process and case study. We highlight feedback from our case study and implications from our observations, and lay out open challenges for visual analytics for forensic cyber security operations, including new opportunities for research and practice.

5.1 Expert Feedback and Implications

In our case study, both analysts had a positive impression of PunyVis, with P_1 commenting that it is an "awesome tool" and P_2 noting that it "would serve analysts [workloads]" for finding homograph IOCs. At the same time, their feedback helped us identify two additional design goals to refine PunyVis and similar visual analytics systems for forensic analysis:

1. Provide scaffolding features that help analysts begin, maintain, and revisit investigation activities, especially for novice users.
2. Support multiple analysis styles for investigation.

Scaffolding for exploration. First, we identified the need for the visual analytics system to provide starting points in the user interface that are clearly targeted at application-specific analysis questions. For example, the analysts indicated that PunyVis' one-click drill-down features were in line with what questions they would try to answer first in real investigations of homograph phishing attacks. We expect that as analysts become more familiar with the tool or even become 'power users', tool support for a finding a starting point will be less important than it is for users new to the tool.

Other types of workflow support could benefit experienced users as well as novices. For example, the PunyVis frame analyzer lets analysts save partial results in the form of subsets of data selections to revisit later. Because even domain and tool experts have limited working memory, features that let analysts suspend their investigation without losing all progress could be impactful. Through user feedback we identified other features like these that could improve workflows, like enabling follow-up annotations for records ("These domains are probably benign because...") and tracking user hypotheses.

Analysis styles and strategies. We found evidence that analysts can have diverging styles for approaching open-ended analysis tasks. This follows past studies that showed domain experts can use a visual analysis tool for the same task in different ways (e.g., [23]), possibly in ways that support each individual's mental model of the problem or data. A related individual difference with possible design implications is the analyst's trust of the underlying mechanisms of the analytic and interface. Sacha et al. recommend improving trust in the system by supporting accessibility of its features [17]. However, accessible applications for a novice (e.g., easy to use interactions) might have a contrary effect for an expert who wants finer control. We plan to support a wider range of granularity in how analysts can interact with the tool in later design iterations. It is possible that how much access is needed to trust the system could be related to measurable human factors like one's cognitive reflection [10], which could be a basis for customizing the view in a tool like PunyVis. A previous study found that this attribute was correlated with whether a person is an "explanation fiend" or an "explanation foe", who tends to prefer more or fewer mechanistic details respectively during decision processes [9].

5.2 Challenges and Opportunities

Extending PunyVis functionality. Next steps for the PunyVis project include incorporating novel features and usability improvements based on the feedback from our analysts. For example, the search feature could be expanded to handle more complex Boolean queries, and the log inspector could be enriched with even more useful metadata, such as information about the domain name registrar and registrant. The experts also sought to view what data was actually sent through request by looking at packet capture data, and requested built-in links to common IT security tools (SIEMs), such as IBM QRadar or Splunk.

Quantitative evaluation of analysis workflows. While our case study led to insights about how analysts approach the homograph phishing problem and use PunyVis, a controlled user study with more participants is needed to better understand and quantify how PunyVis can improve analyst workflows. It is possible that a larger, more diverse study group would help us identify additional analysis styles, or identify additional pain points overlooked during our initial design and evaluation process.

Analysts in our study used Microsoft Excel as a familiar general-purpose analysis tool with filterable, tabular views in contrast with the PunyVis system. Expanding the evaluation to include existing visualization plug-in frameworks for SIEM tools, like Splunk and Kibana dashboards, might uncover other insights for how analysts search for IOCs. Additionally, these plug-in frameworks could provide an opportunity to integrate elements of the PunyVis design, like the Unicode homograph tree, into existing security toolkits.

Support for non-English-language victims. One challenge facing PunyVis is that it is tailored to help identify phishing attempts against English readers. We had limited access to the test data and user base needed to address the problem for other language scripts beyond Latin characters. However, confusable characters exist in most scripts. We believe the PunyVis algorithm and interface could be extended to adapt to user-selected languages and scripts. Furthermore, a general-purpose system for detecting homograph phishing that works across multiple languages could be useful in helping to protect people and organizations elsewhere in the world. Instead of the simpler approach we show in Figure 3a, the algorithm could instead search for any matches in a language- or site-specific 'top domain names' list, ensuring that IDN homographs of other IDNs would be detectable. Alternatively, the approach by Liu et al., which uses a visual similarity metric, could potentially extend to ideogram languages, provided the analyst has the capability to render and compare all of the logged DNS data [15].

Forensic support for Unicode attacks beyond homographs. Additionally, there are opportunities to research and develop further tools for forensic analysis of Unicode-based phishing methods beyond homograph attacks. A multitude of IDN and Unicode-based phishing types have been identified and outlined [6, 14]. Not all of these are addressed by Unicode and IDN standards. For example, through bidirectional text spoofing, the ordering of displayed characters can let attackers spoof names using scripts that are written right-to-left. The Internet Engineering Task Force has published standards addressing issues with IDNs in right-to-left languages [1], but there still may be a need for forensic evaluation tools to cover such cases. A simple attack using cascading style sheets and custom fonts could be used to obscure the rendering of hyper link text. Finally, non-visual phishing attacks such as the use of similar domain names ("chase.com" and "chasebank.com") present interesting opportunities to generalize PunyVis for more traditional phishing use cases.

6 Conclusion

This paper outlines challenges in the forensic analysis of DNS data to assess the threat of homograph phishing attacks by cyber security analysts. In this attack, internationalized domain names (IDNs) using Unicode characters are crafted to deceive people into visiting unintended URLs that visually resemble other domain names. This type of attack is difficult to identify definitively using automated approaches since there exist predominately legitimate uses of the protocol that enables it, called Punycode. Therefore, human readability and judgment about domain names are important in estimating the likelihood that an IDN is part of a phishing campaign.

To address these challenges, we created PunyVis, a visual analytics system for identifying homograph attacks. PunyVis works by combining an analytic that helps rank the data with an interface that lets an analyst use these rankings to investigate the most suspicious records. Specifically, it includes:

- A rule-based algorithm for ranking and categorizing Punycode-encoded domain names by risk, and
- An interactive visualization tool with multiple components tailored towards helping analysts find and evaluate instances of homograph attacks. The tool includes a novel visualization for domain names that we call a *Unicode homograph tree*, which shows relationships between visually-confusable characters in IDNs.

We received feedback on the design of PunyVis as well as insights about analysis workflows from cyber security analysts with experience in forensic threat assessment. We observed divergent strategies for using the tool to hunt for suspicious domain names, suggesting that our tool and others must support different modes of investigation that are user-dependent. We have no reason to believe there is only one correct way to explore these data and suggest interfaces like PunyVis must be flexible in order to support sustained adoption by users. We also found that some of the most appreciated features by the analysts were ones that enabled them to quickly start and resume analysis activities without having to browse massive logs of DNS records with ad hoc methods. These features help people divide and conquer problem spaces that may be otherwise intractable.

Systems like PunyVis have to potential to improve analysts' workflows by providing and preserving additional context when exploring large datasets, like DNS server logs. In cyber security this approach is particularly effective, because evidence of an attack campaign may be spread across time or targeted at many victims. We believe visual analytics that support users' ability to pivot across the data efficiently as they explore multiple hypotheses are critical in helping analysts recognize, and ultimately mitigate, homograph phishing and other threats.

References

[1] H. Alvestrand and C. Karp. Right-to-left scripts for internationalized domain names for applications (IDNA). RFC 5893, Internet Engineering Task Force (IETF), August 2010.

[2] Amazon.com. Alexa top sites. `https://www.alexa.com/topsites`. [Accessed: 2019-06-19].

[3] The Zeek network security monitor. `https://www.zeek.org`. [Accessed: 2019-05-14].

[4] A. M. Costello. Punycode: A Bootstring encoding of Unicode for Internationalized Domain Names in Applications (IDNA) . RFC 3492, Internet Engineering Task Force (IETF), March 2003.

[5] A. M. Costello. Internationalized Domain Names for Applications (IDNA): Definitions and Document Framework . RFC 5890, Internet Engineering Task Force (IETF), August 2010.

[6] M. Davis and M. Suignard. UTR 36: Unicode security considerations. Technical report, Unicode Consortium, Aug 2014.

[7] R. Dhamija, J. D. Tygar, and M. Hearst. Why phishing works. In *Proceedings of the SIGCHI conference on Human Factors in computing systems*, pp. 581–590. ACM, 2006.

[8] P. Faltstrom, P. Hoffman, and A. Costello. Internationalizing Domain Names in Applications (IDNA) . RFC 3490, Internet Engineering Task Force (IETF), March 2003.

[9] P. M. Fernbach, S. A. Sloman, R. S. Louis, and J. N. Shube. Explanation fiends and foes: How mechanistic detail determines understanding and preference. *Journal of Consumer Research*, 39(5):1115–1131, 2012.

[10] S. Frederick. Cognitive reflection and decision making. *Journal of Economic perspectives*, 19(4):25–42, 2005.

[11] E. Gabrilovich and A. Gontmakher. The homograph attack. *Communications of the ACM*, 45(2):128, 2002.

[12] IDN in Google Chrome. `https://www.chromium.org/developers/design-documents/idn-in-google-chrome#TOC-Google-Chrome-s-IDN-policy`. [Accessed: 2019-06-04].

[13] P. Hoffman and M. Blanchet. Nameprep: A Stringprep Profile for Internationalized Domain Names (IDN) . RFC 3491, Internet Engineering Task Force (IETF), March 2003.

[14] V. Krammer. Phishing defense against IDN address spoofing attacks. In *Proceedings of the 2006 International Conference on Privacy, Security and Trust: Bridge the Gap Between PST Technologies and Business Services*, p. 32. ACM, 2006.

[15] B. Liu, C. Lu, Z. Li, Y. Liu, H. Duan, S. Hao, and Z. Zhang. A reexamination of internationalized domain names: The good, the bad and the ugly. In *2018 48th Annual IEEE/IFIP International Conference on Dependable Systems and Networks (DSN)*, pp. 654–665, June 2018. doi: 10.1109/DSN.2018.00072

[16] IDN display algorithm. `https://wiki.mozilla.org/IDN_Display_Algorithm`. [Accessed: 2019-06-04].

[17] D. Sacha, H. Senaratne, B. C. Kwon, G. Ellis, and D. A. Keim. The role of uncertainty, awareness, and trust in visual analytics. *IEEE Transactions on Visualization and Computer Graphics*, 22(1):240–249, 2015.

[18] The Unicode standard: A technical introduction. `https://www.unicode.org/standard/principles.html`. [Accessed: 2019-06-04].

[19] L. Von Ahn, B. Maurer, C. McMillen, D. Abraham, and M. Blum. reCAPTCHA: Human-based character recognition via web security measures. *Science*, 321(5895):1465–1468, 2008.

[20] L. Wenyin, A. Y. Fu, and X. Deng. Exposing homograph obfuscation intentions by coloring unicode strings. In *Asia-Pacific Web Conference*, pp. 275–286. Springer, 2008.

[21] G. Ye, Z. Tang, D. Fang, Z. Zhu, Y. Feng, P. Xu, X. Chen, and Z. Wang. Yet another text captcha solver: A generative adversarial network based approach. In *Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security*, pp. 332–348. ACM, 2018.

[22] X. Zheng. Phishing with unicode domains. `https://www.xudongz.com/blog/2017/idn-phishing/`. [Accessed: 2019-05-14].

[23] C. Ziemkiewicz, S. Gomez, and D. Laidlaw. Analysis within and between graphs: Observed user strategies in immunobiology visualization. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pp. 1655–1658. ACM, 2012.