New Top-Level Domains, challenges

David Singer (with others), Apple

August 2015

1 Introduction

1.1 What's happening?

The Internet Corporation for Assigned Numbers and Names (ICANN) is in the process of enabling many new top-level domain names (TLDs) – see http://newgtlds.icann.org/en/. To date, over 500 have been registered (see http://newgtlds.icann.org/en/program-status/statistics), out of nearly 2,000 applications. We expect between 1,000 and 2,000; though a goodly number of those are being registered 'defensively' (i.e. they are not expected to be used) this is not true of all of them.

International domain names (IDNs) were enabled in https://www.icann.org/resources/pages/rfcs-2012-02-25-en. A goodly number of the new top-level domains are or will be international. In some respects, IDNs have stayed isolated since 2003; one doesn't often encounter e.g. a host-name with Chinese domain names in it, outside China. This may well change; and we probably should not assume 'cultural bias' in the use of domain names. Just as we expect latin alphabet domain names to be seen and used world-wide, we should reasonably expect the same of non-latin names – in website address or email addresses – to be increasingly visible outside the regions of the world in which their script is common. Many governments are engaged with using new extensions (e.g., http://be.brussels/about-the-region/the-government-of-the-region and http://www.gov.scot/) and China's Ministry of Industry and Information Technology is making it a requirement that all Chinese government websites transition to Chinese domain names.

This document explores some of the challenges that these introductions may bring.

ICANN has the Universal Acceptance Steering Group, who assume that these new TLDs will or should be generally accepted. However, it seems that at least some of the structures and protocols that use domain names might have assumed that the set of TLDs was small, left-to-right, etc. The technical community may need to think about some of the emerging issues.

1.2 Background

Here are some rough definitions of terms for the purposes of discussion.

Host-name: a series of domain-names separated by period characters (e.g. www.example.com).

Domain-name: a name registered in a domain name server.

Punycode: a way to transform strings that use full Unicode into strings that only use (informally) 'ASCII'.

Directionality: an attribute of text: notably does it read left-to-right or right-to-left. (This is a complex subject dealt with at length in http://www.unicode.org/reports/tr9/).

Structured text: a string such as a hostname, file path, or mail address, that is composed of components separated by separator characters. A URL is two-level structured, having components (host name, file path) that are in turn structured (host names are composed of domain names, file paths of folder and file names, etc.)

Universal Acceptance Steering Group (UASG): In February 2015 the ICANN community formed the UASG to start outlining and prioritizing issues around UA and further divided into four working groups:

- Topline & Technical Issues Project Group: Intends to "identify the issues, come up some solutions, and then pass them to the Community Outreach group to get the message polished and distributed.

- International Project Group: Focusing on the non-ASCII part of the UA issues – in e-mail addresses (EAI), domain names (IDNs) and addresses (IRIs). The International group also expects to develop the issues and then pass to the Community Outreach group for polishing and distributions.

- Measurement & Monitoring Project Group: To come up with some tests for UA Readiness, keeping track of UA Readiness issues identified, and regularly testing and reporting on generic UA readiness. Will provide direction to the Community Outreach team of organisations that have been reported as not-UA compliant:

- Community Outreach Project Group: Tasked with identifying the messages, finding effective communication channels, and then getting the messages into the community. To get input from the other groups about the messages and the audiences.

1.3 Resources and References

- The Register article that discusses at a high level some of the issues anticipated with acceptance of the IDN email address: http://www.theregister.co.uk/2015/01/15/why_you_need_to_rewrite_your_code_for_handling_email/
- Presentation from the Google developer conference last June introducing new gTLDs: https://www.google.com/events/io/schedule/session/22ce27dc-7cbf-e311-b297-00155d5066d7
- Link to ICANN's resource page on Universal Acceptance: https://www.icann.org/resources/pages/universal-acceptance-2012-02-25-en
- ICANN's list of top-level domains https://www.icann.org/resources/pages/tlds-2012-02-25-en
- Unicode Security Considerations, Unicode http://www.unicode.org/reports/tr36/.
- Unicode Security Considerations ('confusables'), Unicode http://unicode.org/reports/tr39/.
- The IETF RFCs are helpfully grouped and referenced by ICANN https://www.icann.org/resources/pages/rfcs-2012-02-25-en. (RFCs IRI: 3987, IDNS: 5890-5895, Email etc.: 6530-6533, 6855-6858,
- ICANN's Universal Acceptance Steering Group (UASG)
  confluence page: https://community.icann.org/pages/viewpage.action?pageId=47255444
  UASG projects group page: https://community.icann.org/display/TUA/Project+Groups

2 Is this a (valid, operating) host name?

Historically, valid hostnames have had more than one domain name, i.e. they always contain a period. In addition, there were few enough top-level domain names, and they changed so rarely, one could detect whether one of the 'historic five' (.com, .mil, .net, .gov, .edu) or a geographic name (generally, an ISO 3166 Country Code) was in use. This is no longer the case.

Some of the top-level domains (.google, .apple, .beats) are owned by a single entity and could be used as host-names in their own right. It may be dangerous to assume that something can only be a host-name if it contains a period.

Some systems use database lists (e.g. Public Suffix) to check whether something is plausibly a host-name.

3 Public Suffix issues

When HTTP servers set cookies, they can request that they be set against a host-name that is 'higher level' than the server is operating at. E.g. images.example.com may set a cookie that is returned to all servers under example.com. It is not legal to set a cookie on the part of a host-name that is public. Mozilla maintains https://publicsuffix.org/ which documents all these public suffixes. This list will need to be updated and consulted perhaps more frequently, and note also that there are now top-level domains that have a null, zero-length, public suffix (.google, .beats, .apple).

4 What domain names are available for private-network/internal use?

It used to be that one could have internal networks and names, resolved by internal DNS, as long as you avoided the historical (.com, .mil etc.) and the country-code TLDs. It is no longer clear what is available (if anything) without clashing with the public namespace.

5 Visual issues – ordering

A domain name may not mix right-to-left and left-to-right scripts (but they can include non-directional characters such as "-" and numerals). Normal text-presentation rules are currently applied to domain names. Since the separator characters ("/", ".", "@" etc.) are non-directional, consecutive runs of names that have the same direction are set en masse in that direction. Example: consider a host-name made from left-to-right domain names ltr1 and ltr2, and right-to-left names rtl1 and rtl2, such that the logical order is ltr1.rtl1.rtl2.ltr2. It will actually present as ltr1.rtl2.rtl1.ltr2.

The readers of right-to-left languages are used to this and expect it, and there are few enough right-to-left languages in common usage (principally Arabic and Hebrew) that readers of one at least recognize the other. However, if these names get exposed to non-readers of right-to-left languages, they may be confused.

In addition, if the last element of a host-name (i.e. a new iTLD) is right-to-left, then it may 'cross over' into the path-name. Consider a host-name that is ltr1.rtl1, and a path name that is /rtl2/index.html. The URL is logically http://ltr1.rtl1/rtl2/index.html but it will display as http://ltr1.rtl2/rtl1/index.html. Similar cross-over may happen round the "#" that starts a URL fragment or "?" that starts a URL query.

Though it may appear desirable always to present structured text in structure order, i.e. not do these reversals, we cannot do this. For a start, those who read right-to-left languages do not expect visual order to follow from structural order, they expect reading order to follow structural order. Secondly, when text is not recognized as structured (which will happen) it will display in this order. Third, introducing this change would increase rather than reduce ambiguity (is the text in reading order or structural order?).

We need to establish much better practices at presenting structured text; for example, in the address bar of the browser one might show only the hostname. Apart from the famous blue underline, we don’t do much today to indicate that something is structured and what its parts are. For example, we could color the hostname and the pathname, the query and the fragment, differently, so if they visually mix we’d see the colors mixing. In addition, we could make it easier to discover what the structural/logical ordering is (e.g. hover could pop something up that shows that www is a subdomain of apple which is in com).

If something is authored to mislead, we can’t expect that users will often pay attention: but we should make it easy for them to do so, and encourage them to do so.

It would probably be good if there were ‘common best practices’ across the industry, to reduce confusion.

6 Visual issues – non-directional characters and 'crossover'

6.1 The issue

A more confusing cross-over can happen with numerals and "-". Consider the following example:

```
% mkdir א
% cd    א
% mkdir 3-6-15-Notes
% cd    3-6-15-Notes
% pwd
/Users/someone/3-6-15/א-Notes
```

With the advent of non-latin (and hence potentially RTL) top-level domains, this becomes a problem not only within hostnames and pathnames, but in a URL, as the final host component is logically next to the first path component. Also, a URL can have a fragment (text after a “#”) or query (text after a “?”). The initial text of those may get visually mixed into the preceding text. Someone might attach a fragment to a URL, knowing that either the local software or the server will discard it (if it doesn’t apply to the resource in this context), but that helps to mislead or confuse.

This is confusing to all readers (even of right-to-left languages). The problem comes because the separator characters, which have special meaning in structured text, are being treated as if they were 'ordinary punctuation'.

We need to stop doing this.

PROPOSAL: Present structured text as if each Component was surrounded by Unicode isolates, i.e. require that components, or parts of them, not visually “cross” the separators. This MUST be done when the structured text is acted on or actionable, or known from context to be structured, e.g. a URL or mail address that has been followed or can be clicked. It SHOULD be done whenever text is recognized as structured.

6.2 Examples

Here are some live examples to illustrate what happens. Consider a hostname whose top-level domain is آبلدستور, and the first path of the pathname is بازار. Put together, with http://, it will read as http://آبلدستور/بازار

But if we now add a subdomain, such as www, it will now read http://www.آبلدستور/بازار ; Note this is as above, the logical order is LTR1, RTL1, RTL2, but the visual order is LTR1, RTL2, RTL1.

Numbers are particularly problematic because they are considered neutral characters. If you have TLD ال and foldername 123apple, typing it in your browser will result in

http://ال/123apple

Though the expected behavior is probably something like

http://123apple/ال

Similar issues can happen with # and ? in URLs. Consider a URL like

http://google.com/ال?1s

google.com is the hostname, the path is ال, and the query string is 1s. But that is not obvious at all from the current visual representation. Presumably, we want something like

http://google.com/ال?‎1s

Note that to force this to draw correctly, we must currently insert an LRM (left-to-right mark) after the question mark to resolve the ambiguity in the directionality of the neutral 1.

Note that the domain name system RFCs (notably RFC 5893) does not allow domain names to start with numerals, so this problem was avoided as long as TLDs were LTR – but it now reappears with the appearance of RTL top-level domains, as we cannot control the first part of the *pathname*.

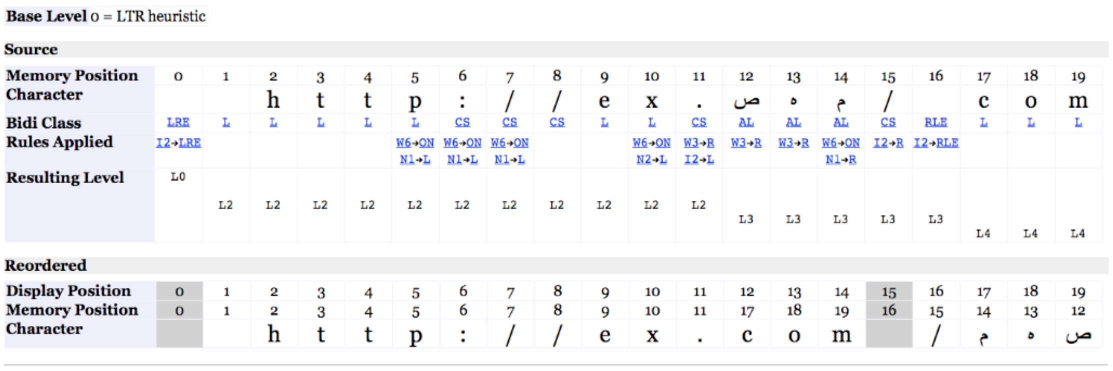
7 Visual issues – spoofing and phishing

There is an ongoing issue of spoofing: people make URLs with zero and the letter 'o' substituted, or the digit 1 and lower-case 'L' (l), etc. This has the effect of letting users think they are visiting Google when in fact it's goog1e. This is known as a homograph attack (http://en.wikipedia.org/wiki/IDN_homograph_attack) and it is extensively discussed, e.g. in Unicode TR #36.\. Another example is a homograph attack where U+30FC (Katakana-Hiragana prolonged sound mark) is substituted with a simple En-dash (U+2013), or vice-versa.

Much worse, however, is the issue that one can construct a URL that has BIDI characters in the path name (folder names), that, combined with a RTL top-level domain, can seriously mislead the user as to what hostname the URL points to. This URL, for example, has an Arabic top-level domain name, but (under some circumstances) it appears to start http://ex.com/

http://ex.م هص/com

Here it is in the in-memory order, showing that someone devious has put a right-left-embedding in the first component of the path (which is actually <rle>com). Note that RLEs are not normally part of names, but we may not be able to control when they are part of a filename (or query, or fragment).



Base Level 0 = LTR heuristic

Source

Memory Position	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19
Character			h	t	t	p	:	/	/	e	x	.	ص	ه	م	/		c	o	m
Bidi Class	LRE	L	L	L	L	L	CS	CS	CS	L	L	CS	AL	AL	AL	CS	RLE	L	L	L
Rules Applied	I2→LRE					W6→ON N1→L	W6→ON N1→L	W6→ON N1→L			W6→ON N2→L	W3→R I2→L	W3→R	W3→R	W6→ON N1→R	I2→R	I2→RLE			
Resulting Level	L0	L2	L2	L2	L2	L2	L2	L2	L2	L2	L2	L2	L3	L3	L3	L3	L3	L4	L4	L4

Reordered

Display Position	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19
Memory Position	0	1	2	3	4	5	6	7	8	9	10	11	17	18	19	16	15	14	13	12
Character			h	t	t	p	:	/	/	e	x	.	c	o	m		/	م	ه	ص

8 Visual issues – ambiguity

Worse still, there is not a one-to-one relationship between the visual representation of BIDI text and the in-memory order: it is possible for two *different* in-memory strings to have the *same* visual representation. Under these circumstances, the poor user actually formally cannot know, based on what they see, where the URL will take them.

9 Visual issues – text ordering

There is another rather subtle problem that the new TLDs make worse. If one forces normally left-to-right text to be presented right-to-left, and a partial URL is presented to the user (i.e. without the scheme prefix, such as http:), it's now possible to construct a URL that looks as if what the user is visiting is one host (but that's actually the file pathname) and the actual file pathname appears to be the host.

Example: if one forces the partial URL lmth.crs/moc.elppa.www to be presented RTL, it appears as www.apple.com/src.html, which looks plausible and innocuous. (It also conveniently conveys the name of the site being phished). This can be done using the Unicode Right-left-ordering (RLLO) character (which doesn't always survive presentation, so isn't used here, but does survive in some contexts, e.g. in email etc.)

In the example above the actual top-level domain is 'crs', which looks plausible as part of a filename when reversed. New TLDs that might look innocuous when reversed and presented as part of a filename, or are palindromic, include:

DOG, LOL, GOLF, CBN, GOOG, REIT, CRS, TUI, DAD, OOO, TOP, SCB, CITIC, FOO, GAL, EUS, MEET, AXA, PARTS, WED, TIPS

10 Recommendations:

1. UTR #36 does not discuss structured text or the possible problems associated with it, and needs to. Similarly there are IETF documents that ought to address this.
2. At the moment, structural separator characters are treated as if they were simply the punctuation they appear to be. This means that when other non-directional text is in the components, text can visually ‘cross over’ the separators, leading to presentation that is misleading even to those who read all the scripts. We need to stop doing this.
3. Document the issue that structured text is presented with the components in reading order, not visual order, and the mitigations and work-arounds.
4. Get much better at presenting structured text so that the reader can tell how it is structured and what the structural ordering is, even if they don't read all the scripts used. Establish some best practices in this area.