IDN and Variants

Presented at the Yeti DNS Workshop

Marc Blanchet
Viagénie
marc.blanchet@viagenie.ca

2016-11-12
Context

• DNS does strict identifier mapping
• Humans using languages and scripts do fuzzy mappings
• Can we reconcile them?
Available Toolkit

• DNS
• IDNA
• LGR
DNS

- DNAME
- DNSBundled BOF this week
IDNA

• Punycode encoding: encodes UTF-8 into restricted ascii for DNS purposes
• IDN labels start with xn--
• IDNA2003:
  • Lists codepoints that are valid
• IDNA2008:
  • Based on unicode properties of the code points, defines which codepoint is valid (PVALID)
  • When a new release of Unicode, a new table of codepoints is computed (automatically).
  • Should be no issue. Right?
• No symbols, emoji, ... sad eh?
IDN Tables

• IDN (IDNA2003) started to be deployed and used. Registries (for second level domain names) had to define which scripts and characters they support.
  • IDN tables
    • First attempt: no standard format definition. Can’t process. Only for documentation purposes
    • http://www.iana.org/domains/idn-tables
  • No real compatibility between IDN tables:
    • For a given language or script, a codepoint may be valid in one TLD and invalid in another
    • U+00E0 (LATIN SMALL LETTER A WITH GRAVE): not supported in .ca, but supported .camera
    • Variant support was variable
Variants

• Many definitions

• Possible variant in my own language:
  • ‘a followed by e’ can be written either as ‘ae’ or ‘æ’

• ICANN set up a committee to identify how to support variants in Root Zone
  • What is a variant: « There is today no fully accepted definition for what may constitute a variant relationship between top-level labels, and the results of the case studies suggest that it would be very difficult to come to a single definition at the current time, because there is more than one phenomenon being discussed. »
Label Generation Rules (LGR)

• One of the Recommendation of the committee
• Together with a procedure to define them (more later)
• To define how to correctly support IDNs (and variants), a DSL (Domain-specific language) was defined to encode the rules of each writing system.
• NOT used in the DNS. Used at the registration time to identify if a specific label is valid, the variants and the validity of the variants
LGR

• LGR is a way to define
  • a repertoire (list of codepoints and sequences of codepoints)
  • relationship between codepoints (variant mappings)
  • rules to be applied to those codepoints
  • rules to be applied to the whole label
• LGR may be defined for languages and/or scripts
• Has the potential to be used in other domains than DNS
LGR Specification

• Initially defined by Kim Davies
• IETF wg: lager
• RFC7940
• XML
• Next slides to show a glimpse of what one can do by using the LGR specification
  • Examples taken from « real » submitted LGR of various scripts
LGR Example: Meta

- `<?xml version="1.0"?>`
- `<lgr xmlns="urn:ietf:params:xml:ns:lgr-1.0">
  - `<meta>`
  - `<version>1.0</version>`
  - `<date>2015-11-05</date>`
  - `<language>und-Armn</language>`
    - # xml tag is « language » but can be used to identify a language or script.
    - # An LGR may have multiple <language> tags.
  - `<scope type="domain">.</scope>`
    - # an LGR for .ca would be `<scope type=" domain ">ca.</scope>`
  - `<unicode-version>6.3.0</unicode-version>`
  - `<description type="text/html"> ..</description>`
  - `<references>...</references>`
- `</meta>`
LGR Example: Repertoire

- <data>
- <char cp="0586" tag="sc:Armn" ref="0 100" />
  - # cp = codepoint, unicode U+0586 = ﶯ
  - # tag = identifier of a set of codepoints
  - # ref = reference number (for documentation purposes)
- <range first-cp="0583" last-cp="0584" tag="sc:Armn" ref="0 100" />
  - # defines a range of codepoints
- <char cp="0061 0065"/>
  - A sequence of codepoints: “a e”
LGR Example: Variants

- `<char cp="0649" tag="sc:Arab" ref="0 100">`
- `<var cp="0626" type="blocked" />`
- `<var cp="064A" type="blocked" />`
- `<var cp="067B" type="blocked" />`
- `<var cp="06CC" type="blocked" />`
- `<var cp="06CD" type="blocked" />`
- `<var cp="06D0" type="blocked" />`
- `<var cp="06D2" type="blocked" />`
- `</char>`
- `# from Arabic LGR`
- `٢،٣،٤،٥،٦،٧،٨،٩،٠`
LGR Example: Variants

• ۓ ڪ ڪ; ڪ; ڪ; ڪ; ڪ; ڪ; ڪ;
• From the script perspective, all these codepoints are equivalent: a user may use any of them to express
• But each one may be used in specific languages
• An owner of a label containing one of these want to:
  • either own or reserve the other alternatives of the label
  • Or disable someone else to own/use/register one of the alternative label
LGR Example: Variants

• Arabic is a script where many codepoints have many variants.

• If all variants of each codepoint (that has variants) are allocatable (i.e. can be used in the DNS), then the number of allocatable variant labels can be large:
  • A label of 4 codepoints, each one having 5 variants generates <quiz> labels, all « equivalent »!
LGR Example: Rules

- `<char cp="17B6" when="follows-B-subscript-consonant-and-depvowel" tag="dependent-vowel" ref="3 100 102" />
  - # context defined by the « when » rule. (actual rule defined in the <rules> section
LGR Example: Rules

• <rule name="no-mix-kaf-with-ring-keheh-with-three-dots-above" comment="do not mix Arabic letters KAF WITH RING and KEHEH WITH THREE DOTS ABOVE">
  • <choice>
    • <rule>
      • <char cp="06AB"/>
      • <any count="0+"/>
      • <char cp="0763"/>
    • </rule>
    • <rule>
      • <char cp="0763"/>
      • <any count="0+"/>
      • <char cp="06AB"/>
    • </rule>
  • </choice>
• </rule>
LGR Example: Rules

- `<action disp="invalid" match="no-mix-kaf-with-ring-keheh-with-three-dots-above" comment="do not mix Arabic letters KAF WITH RING and KEHEH WITH THREE DOTS ABOVE" />

- # if a label matches the rule defined, then the disposition of this label is “invalid”. i.e. cannot be used/allocated/assigned/...
Defining Script LGRs

• One of the difficulties is where to draw the line for encoding the script rules
  • It should not embed the grammar. Reminder: DNS is just identifiers, mnemonics. The fact that in a script/language, one cannot use three consonants in a row is _not_ a problem for mnemonics. i.e. ‘wdtrgfjk’ is a valid label.

• Each script/script family has his own specifics

• Some scripts have different Unicode encoding models
End Result

• A label is identified by its script
• The appropriate script LGR is then processed and applied to the label by an LGR processor
• The output can be:
  • allocatable
    • The label is conformant to the LGR and therefore can be assigned and delegated
  • Blocked
    • The label is blocked by some rules of the LGR and therefore can not be assigned nor delegated
  • Invalid
    • The label contains invalid codepoints or …
LGR Definition and Integration Process

• ICANN is currently handling a process from a well-defined procedure where:
  • Each script community get together (Generation Panel) to create and submit their script LGR.
  • These proposals are sent to public comments
  • An Integration panel of experts (5) validate and integrate the various submitted script LGRs into an integrated one that defines the Root Zone LGR
• In parallel, second level LGR are also defined, so that registries may use these proposed ones (which means more coherence globally) or define their own
Conclusion

• DNS is a strict identifier mapping. Languages and scripts are not.
• DNS has no variant mapping mechanism.
• Suggestion: attend DNSBundled BOF
• LGR is a formal specification of how to validate a label for each script.
• LGRs for each script are being defined and integrated
• LGR is used at the registration level, not in the DNS
• Defining LGRs is difficult, given the fuzzy nature of languages and scripts.
• Opportunity to Yeti, as a testbed, to try things?
Thank You. Questions?

• References:
  • Lgr-toolset
    • Opensource and as a VM to use
    • Python cmd-line and web interface
    • https://www.icann.org/resources/pages/lgr-toolset-2015-06-21-en

• Marc Blanchet, marc.blanchet@viagenie.ca