



Implementing an IDN Registry

Jon Lawrence
AusRegistry International
APTLD Meeting - Beijing
20th August 2009



AusRegistry International

- Based in Melbourne, Australia
 - Founded in 1999
 - ICANN Accredited Registrar since 2000
 - .au Registry Operator since 2002
- Domain Name Registry Services
 - Registry Systems and Software Provider
 - Consultancy and Training services
 - Working with ccTLD managers and new TLD applicants



Experience

- Australia
 - .au ccTLD Domain Registry Operator (2002)
 - under contract until 2014
 - .au ENUM Registry Trial (2005)
 - Australian REC Registry (2005)
- Registry Software & Consultancy Services
 - United Arab Emirates (2006)
 - .ae ccTLD
 - IDN ccTLD Fast Track: امارات.
 - TBA (2009)
 - .ASCII & IDN ccTLD



Agenda

- Basic implementations
 - Risks of a basic implementation
- Responsible implementations
 - IDN specific policy
 - Handling variants
 - Implications of implementing IDNs



Basic implementation



Basic implementation

- Allow “xn--” registrations
- Request and store a language tag
- Ensure it is protocol valid
- Verify that the Unicode code points it maps to are in a restricted set that applies to the language in use
- All other operations work on “xn--” form only



Risks of a basic implementations

- Significant end-user, Registrant and Registrar confusion
 - Higher support costs
 - Lack of acceptance
- Policy risks
 - No control over offensive or illegal registrations
- Security risks
 - Phishing and other misdirection style attacks
- Reputational risks
 - Loss of confidence in the namespace



Responsible IDN implementation



IDN-specific policies



Developing IDN-specific policies

- IDNA only describes
 - Allowable Unicode code points
 - Broad validity rules
 - BIDI (Bi-Directional) rules for string stability
 - Use of Unicode Normalisation rules (NFC)



Developing IDN-specific policies

- Local policy therefore needs to define
 - Unicode code points that make up your language(s)
 - What code points or sequence of code points you consider to be variants of each other
 - What mappings (if any) you would like application developers / Registrars to apply
 - Reserved terms



Handling variants



Handling variants

- Blocking – preventing registration
- Bundling – enabling or even forcing ‘linked’ registrations



Blocking



The variant generation method

- Treating two characters as variants of each other

$\acute{e} = e$ $e = \acute{e}$ $\text{café.com} = \text{cafe.com}$
 $\text{`} = 1$ $1 = \text{`}$ $\text{```.ae} = \text{111.ae}$

- Calculating variants
 - Can be performed on input to all commands
 - or*
 - All variants can be stored at time of registration for later comparison



The Canonical method

- Assign each character a canonical form
 - The 'base' form of the character
 - e -> e
 - é -> e
- Perform a simple substitution of each character for its canonical equivalent
 - Generates the canonical form
 - cafe.com -> cafe.com
 - café.com -> cafe.com



The Canonical method

- Define these canonical mappings for all characters in the zone
- Only storing one additional version of each domain (the canonical form)
 - Avoids the potential performance and storage overheads of the variant generation method
 - Simple algorithm, less error prone, easy to optimise



Bundling



Bundling

- Provisioning variants
café.com + cafe.com
- Many ways to implement this and many considerations



Bundling considerations

- Mandatory or optional?

- Other impacts
 - DNSSEC
 - Charging model
 - Accounting and reporting
 - Is a bundle of three domains one registration or three?
 - Performance impacts
 - Mutual exclusion



Implications of implementing IDNs



Registry Website & other Interfaces

- Registry 'Human' interfaces also have to be updated
 - Apply local mappings, etc.
- WHOIS (web based and port 43)
 - xn-- and user form?
 - Unicode in response (UTF8?)
- Registry website
 - Searching and other Lookup functions
 - Multilingual help Files
- Accounting and Reporting Interfaces
- Zone tools



Registry - Registrar Protocol

- EPP doesn't natively support IDNs
 - Language tag
 - Only one domain field
 - » IDN protocol recommends you should get both
 - ASCII form: xn--dabnbdasf. xn--pbknsdw
 - Script form: КОМПАНИЯ.РФ
- New error & poll messages
- Management of bundles



Registrars, Registrants and end users

- Registrars now have a much harder job to do
 - Interpret what the Registrant wants
 - Turn that into something protocol valid (to map or not to map?)
 - Explain all of this to the Registrant

- Education campaigns
 - Ensure consistent messaging

- Registrar tools
 - Meaningful error messages
 - Onscreen keyboards



Effects on DNS

- Allowing or enforcing bundles will result in an increase in the size of the zone file
 - Significant additional cost if DNS is outsourced
- Zone Management
 - May require new tools & techniques
- DNSSEC considerations
 - Bundling
 - DNAMES versus Delegations



Security considerations

- Punycode overloading
- Punycode reverse engineering
 - xn--trademark or xn--rude-or-offensive-word
- Variant overflow (deliberate or accidental)
 - Some really big numbers!
- Phishing and other scams
 - Cyrillic “a” versus ASCII “a”, etc.
- Supplementary characters
 - Java issues



System performance impacts

- Validation rules and cross checking
- Many ASCII optimisations are invalidated
- Performance hit even on non-IDN domains
 - Certain checks still need to be performed



Summary

- Complex & difficult but important
- Significant risks if not done properly
- There is help out there if you need it!

jon.lawrence@ausregistry.com

www.ausregistry.com



Questions?

jon.lawrence@ausregistry.com

www.ausregistry.com



