Internationalization Considerations for CRISP

Prepared by Eric A. Hall

on behalf of Verisign Labs

May 9, 2003

Table of Contents

1. Overview and Introduction

This report is divided into two man sections, each of which deal with the two specific proposals currently in front of the CRISP working group. Each of those sections are further sub-divided into three subordinate sections.

The first sub-section identifies the domain name data-types which are reused throughout the specifications. The second section identifies those places where domain names are provided as data, typically using one of the well-known data-types, although other instances are also discussed (such as email addresses, which contain domain names but which do not typically use the domain name data-type rules). Finally, the third section discusses other kinds of data which are likely to be subject to localization and internationalization considerations, such as date strings, contact information, and so forth. In each of these areas, the ramifications of internationalized data are discussed, and recommendations for the proper handling of these sequences are made.

There are two principle external sources which have directly influenced these recommendations. First among these are the IDNA collection of specifications, which describe a technique for encoding internationalized domain names into an ASCII-compatible sequence. Since these sequences will often be used for domain names in general, they will also be used for various operations which either accept domain names as input, or which generate domain names as output. The second external influence in these recommendations is RFC 2277, which defines the IETF policy on charsets and languages in Internet applications.

The two proposals also have different usage environments, and this also affects the recommendations. For example, the LDAP-WHOIS service is intended to be used in conjunction with other LDAP applications, meaning that the domain name data will be directly visible in a variety of scenarios beyond that specific service. Conversely, IRIS defines an application-specific transfer format and does not govern repository views, meaning that the domain names which are used for IRIS can be made efficient for that service in particular. In some cases, this means that a decision which is appropriate to one of the proposed services may not be appropriate for the other.

2. LDAP-WHOIS Issues

The LDAP-WHOIS specifications suffer from having to share data with other kinds of LDAP applications. Specifically, LDAP-WHOIS data is expected to be reused for a variety of user-facing applications, which means that most of the data should be provided in its "raw" internationalized form wherever possible. Although the LDAP protocol allows this, there are some large holes in this support, making it impossible to offer raw forms of the internationalized domain names everywhere at the current time.

2.1 LDAP-WHOIS Domain Name Data-Types

LDAP-WHOIS currently uses two data-types for the majority of its domain names. These are the well-known and standards-track domainComponent attribute, and the inetDnsDomainSyntax rules which are provided in the LDAP-WHOIS specificatons.

2.1.1 The domainComponent Attribute

The LDAP-WHOIS specifications have several rules which govern the naming of directory trees and entries within those trees. Many of these rules make use of the domainComponent (dc=) attribute defined in RFC 2247, which maps DNS domain names to LDAP attributes in order to form a distinguished name for a directory tree.

These distinguished names are widely used in the LDAP-WHOIS service. In the simplest case, these names act as basic representations of a specific storage pool within a server. In the most complex case, a sequence of domainComponent relative distinguished names may be dynamically added or removed from a search base while the client attempts to locate a server which can process a given query. Because of this variety, the domainComponent attribute is the most important aspect of any effort towards supporting internationalized domain names in the LDAP-WHOIS service.

At the current time, the domainComponent attribute uses the IA5string LDAP syntax rules, which are specifically limited to seven-bit characters. Technically, any seven-bit character set may be used with IA5string (and this specifically includes multi-page character sets with defined ISO-2022 escape sequences), although these sequences are typically interpreted as US-ASCII. However, not all of the coded character sets used throughout the world have seven-bit representations, meaning that it is not currently possible to encode every potential internationalized domain name into an IA5string sequence (there is also ample argument that attempting to do so would be foolish, given the de facto behavior is to interpret IA5string sequences as simple US-ASCII).

Unfortunately, at this time there is not an equivalent to domainComponent which is capable of supporting rich character sets. Although there have been discussions on this subject, no material progress on this front has been made as of yet. As a result, the LDAP-WHOIS service is effectively restricted to the domainComponent attribute and the associated IA5string syntax.

Cumulatively, this means that wherever a domainComponent attribute is used within LDAP-WHOIS, the IDNA encoding of the underlying DNS domain name must be used, with the resulting sequence being stored in the domainComponent attribute.

There are some areas where this choice is problematic. For example, operators will have to use the IDNA sequences for the portions of the hierarchy under their control, even though most operators would likely prefer to reference their portion of the hierarchy by the "natural" form of the domain name, rather than an encoded form of that name. It is

even somewhat likely that some operators will rebel at this requirement, possibly by forcing the use of an alternative "compatible" character set or using a character set which is not technically compatible with the IA5string syntax.

Another consideration in this area is with LDAP-WHOIS clients that want to support internationalized domain names as query input. Since the input domain name has to be mapped to domainComponent labels in order for the LDAP query to function properly, the requirement to use IDNA for this function means that LDAP-WHOIS clients must perform the IDNA encoding as part of this process. In truth, similar conversion efforts would be necessary in other scenarios, so this is not a unique detriment (for example, if the LDAP-WHOIS service supported internationalized domainComponent attributes, conversion would also be necessary if a user provided an IDNA sequence as the search input, with the input being converted into the internationalized form before the query could be submitted).

In the long run, the best solution would be to define and use an attribute which directly supports a richer character set or encoding (at best, one that uses UTF-8). However, even this approach has some ramifications, since not all of the protocols and services which make use of LDAP are necessarily capable of supporting a richer syntax, and those systems may be broken by such a syntax. Until these issues are resolved, the use of IDNA with the domainComponent attribute is required.

2.1.2 The inetDnsDomainSyntax Syntax

In those places where the LDAP-WHOIS specifications explicitly identify a unit of data as a domain name, that data is required to conform to the inetDnsDomainSyntax rules. This syntax is used throughout the LDAP-WHOIS specifications, including user-supplied query input, assertion values in LDAP query messages, commonName attributes associated with entries in server data-stores, URLs, and more.

The inetDnsDomainSyntax rules currently allow eight-bit data, although there are specific rules which limit the kinds and quantity of characters that can be provided. These rules are provided with the intention of mimicking the rules which govern domain names in the Domain Name System directly (specifically, these rules describe escaping rules for problematic character codes, the length of a domain name and its constituent labels, and so forth).

Since this syntax allows for eight-bit characters, it is arguably capable of handling internationalized domain names directly, although there are several arguments against such a strategy. One such argument is simplification; the current recommendation for handling the domainComponent element favors the use of IDNA, and it would be simplest for all of the parties if this same encoding were used wherever any domain name appears (note that the use of an internationalized equivalent to domainComponent would mostly reverse this argument).

Another argument in favor of the use of IDNA is that there are no operational rules for the handling of internationalized domain names as raw data; topics such as escape syntaxes for prohibited characters have not yet been addressed in a form which is suitable for cloning into an LDAP syntax. Without these rules, a client cannot knowingly reject malformed local input, and instead must apply post-input processing in order to detect input errors.

On the other hand, requiring the use of IDNA in inetDnsDomainSyntax strings complicates some other aspects of the LDAP-WHOIS service. The most notable of these complications is with sub-string matches. For example, IDNA encodes "föo" differently than it encodes "föobar", which in turn means that servers would be required to perform decoding on-the-fly whenever a sub-string comparison operation was requested (with the decoded assertion value being compared to decoded commonName attributes in the data-store), which would require greater amounts of server processing.

Another downside with using IDNA is that the entries would not make use of user-friendly domain names. In fact, this is even more of a concern than the equivalent problem with domainComponent, since most operators will only have entries at a registration server, and will not manage their own local servers and directory trees. As such, IDNA equivalent representations of commonName attributes for entries will be much more visible, and non-internationalized versions of their domain names will be more annoying to the operators of those domains.

It is worth noting that both methods will require client-side manipulation of the query input strings, so neither choice is any better than the other in this regard. In an IDNA-centric model, if a user enters an internationalized domain name as raw data then the client would have to convert the domain name to IDNA before the query could be submitted. Meanwhile, a model which focused on the internationalized form would require that the client also perform conversion whenever a user entered an IDNA-encoded domain name as the query input.

As can be seen, both approaches have costs and bonuses, with neither approach having a compelling argument over the other. In the long run, however, it seems that best approach would be to have the inetDnsDomainSyntax rules accommodate internationalized domain names directly. This model will allow for the use of internationalized domain names as commonName values in the directory tree, and as assertion values, and will help to position the LDAP-WHOIS service for use with an internationalized version of the domainComponent attribute should one emerge. Furthermore, the raw internationalized domain names are simpler to compare, without requiring servers to decode and encode domain names for comparison purposes.

Unfortunately, there are no rules which can be readily adopted for this position to be immediately embraced. As a workaround to this problem, internationalized domain names should be required to undergo a round-trip conversion to IDNA and UTF-8 before they are used, thereby ensuring that the entered data is normalized and valid before it is

written to storage or transferred in a message. Note that this process is already partially necessary, so this approach only introduces a minor amount of additional labor.

Note that there will be some instances where internationalized domain names are still required to be provided as IDNA sequences, due to data-typing restrictions. Most of these issues are discussed in more detail below.

2.2 LDAP-WHOIS Domain Name Instances

Domain names are used throughout the LDAP-WHOIS specifications, including user-side input, protocol messages, query processing, referrals, and more. Each of these instances are described in the remainder of this section.

2.2.1 Query Input

In theory, there will be two common methods for entering domain names as query strings. On the one hand, users with sufficiently capable operating systems and client applications will likely enter internationalized domain names in their raw form, since that is how the domain names should eventually appear. On the other hand, users with limited charset support will usually be required to enter an internationalized domain name as an IDNA encoded sequence. Both of these input forms should be accommodated by the LDAP-WHOIS specifications and client applications, with the input domain names being converted to IDNA and back to UTF-8 in order to ensure that a normalized and valid domain name has been provided.

The query input is eventually used to form an assertion value, a search base for the LDAP query, and the SRV resource record bootstrap lookups. These topics are discussed in more detail below.

2.2.2 Assertion Values

The assertion value is what eventually gets compared to the commonName attribute values of LDAP entries in the servers. The assertion value should be the internationalized domain name which results from any conversion of the domain name provided as the user-supplied query string.

2.2.3 Search-Base Sequences

The user-supplied query string is converted into domainComponenet sequences to form a search-base for the query. These sequences are generated dynamically, with the user-supplied domain name being broken into constituent labels, and with each label being mapped to a domainComponent attribute.

As was discussed in section 2.1.1, the domainComponent attribute must currently be restricted to IDNA encoded representations. This requirement also governs the domainComponent attributes which are used to form the search-base.

2.2.4 SRV Processing

A critical element of the LDAP-WHOIS model is the dynamic construction of DNS queries for any SRV resource records which may be associated with the domain name provided in the query input. In this model, the user-supplied domain name is used to construct a DNS query for the SRV resource records associated with that domain name or one of its delegation parents, with the resulting answers telling the LDAP-WHOIS client where to send the LDAP query for the resource in question.

There are two critical domain names listed in this process, which are the domain names used in the Question Section of the dynamically-formed SRV lookup, and the owner domain name of the SRV resource records which form the answers to these queries.

At the current time, the Domain Name System does not support internationalized domain names as raw data (note that this is mostly an operational restriction rather than a technical restriction). As such, the domain names which will be returned in any SRV responses must be restricted to the IDNA form. Subsequently, in order to ensure that the DNS matching performs as expected, the domain names which are dynamically formed in the Question section of the SRV queries must also use the IDNA form.

Cumulatively, this means that the dynamic conversion of LDAP-WHOIS query input into DNS SRV lookups must result in IDNA domain names.

2.2.5 commonName and inetDnsDomainMatch

Once a server has been located and the assertion value and search-base strings have been passed to it, the server has to compare the assertion value to all of the entries with an inetDnsDomain object class in the specified directory tree, with the comparison specifically looking for entries with a commonName value that is clearly superior to the domain name provided in the assertion value.

In order for this comparison to perform with the minimal amount of effort, and in order to ensure that the entries are stored in their friendliest form, it is necessary that the commonName attribute of the entries also use the internationalized domain name form, rather than being stored as IDNA sequences. In this model, any conversion to and from IDNA must take place at the client or at the database-management application, and must not take place at the server itself.

Note that the assertion values are provided with the inetDnsDomainMatch extensible match rule, which must also be extended to allow for internationalized domain names to be presented in their natural form. Since this matching rule uses inetDnsDomainSyntax, the direct extension of the underlying syntax rules should be sufficient.

2.2.6 Email Addresses

Email addresses appear in two distinct usage scenarios in the LDAP-WHOIS collection of specifications, each of which have different ramifications.

First of all, some object classes provide the "mail" attribute as defined in RFC 2798, which strictly defines these values as case-neutral IA5 strings. Due to the limitations of the syntax rules, these attributes are required to use the IDNA encoding.

The secondary scenario involves embedding email addresses within a fully-qualified directory string as part of a contact-lookup mechanism. These instances are not restricted to the IA5 charset rules, and are fully capable of being stored as UTF-8 sequences. These attributes should make use of raw internationalized domain names in order to facilitate simpler comparison operations, better user-friendliness, and other advantages which come from using internationalized domain names in their natural form.

2.2.7 Nameserver Records

Another attribute (or set of attributes) which will have to be given consideration is nameserver lists and queries, whereby the list of authoritative DNS servers for a domain may be displayed and/or queried in various ways. At the current time, there are no authoritative LDAP syntaxes for the listing or query of DNS nameservers (this subject is being explored and developed on a parallel track to this report).

As with many of the other data-types, there are relatively even arguments for and against both choices. The immediate preference is to store and display resource records in the format in which they will be most commonly used, which would be IDNA sequence. Furthermore, there is not a compelling argument for displaying the domain names to the users in their raw form; the people who need to view and work with these domain names will want to see them as they appear in the DNS, and not as they would appear on a billboard or television advertisement, or some other general-consumer medium.

However, this position only forestalls the inevitability of internationalized domain names within the DNS itself, and failure to embrace the inevitable now will almost certainly translate into ten times the work later. Furthermore, since a preference has been shown for "raw" internationalized domain names in other areas, it makes sense to minimize the chaos by echoing that preference for this syntax as well, at least in the hopes of eventually being able to support a single encoding throughout the service. For these

reasons, the direct support for internationalized domain names in their raw form should be allowed and encouraged.

2.2.8 LDAP-WHOIS URLs

The LDAP-WHOIS specifications make frequent use of URLs, either for the purpose of generating and processing LDAP referrals, or for providing generic reference pointers to "more information."

These URLs are normally passed as data inside the labeledURI attribute, which is defined by RFC 2079. The labeledURI attribute is a two-part syntax, containing a structured URI followed by (optional) unstructured text. The entire sequence uses the directoryString syntax, which is essentially UTF-8. However, the structured URI portion of the syntax is required to conform to URI formatting rules, as defined by RFC 2396 (and as extended in protocol-specific URL definitions), with only the unstructured text being allowed to use raw UTF-8. Meanwhile, the protocol-specific LDAP URL (which is one type of URL that may be passed inside of a labeledURI attribute) is currently defined in RFC 2255, and has its own (strict) formatting rules.

Cumulatively, these rules heavily constrain the kinds of character data which may be passed as URLs in the LDAP-WHOIS service.

There are three areas where an internationalized domain name may appear within the LDAP-WHOIS URL: the hostname of a target LDAP server or service provider; the internationalized domain name which was being queried for (using the commonName equivalent of the input); and the distinguished name of the target directory tree (using the domainComponent mapping).

The following example shows these components in a continuation reference referral:

```
ldap://host.example.com/cn=example.com,cn=inetResources,dc=example,dc=com
```

where "host.example.com" is the target server, "cn=example.com" is the relative distinguished name of an entry for the "example.com" domain name which will get converted into an assertion value in the follow-up query, while "dc=example,dc=com" indicates the root of the directory tree on the target server which will become the search base of the new query.

Each of these elements have their own specific considerations, although these considerations are already covered under the rules which govern these URLs.

The domain name of the target server or provider must use a domain name which can be resolved through DNS, which essentially requires IDNA to be used for this element. There are no special considerations with the LDAP-WHOIS service in particular with regards to this element.

The relative distinguished name element which refers to an entry is slightly more complicated, since this document suggests that those entries should be stored as raw UTF-8. The governing specifications already define the appropriate behavior here, which is to encode these sequences as URL-safe strings, using the percent-hack method. However, in order to avoid any unnecessary confusion, the LDAP-WHOIS specifications should state that these sequences are to be treated as UTF-8 sequences which have been percent-hacked for URL transport, and that the decoded sequences must be used for all LDAP-specific functions and operations.

Meanwhile, this document also endorses the short-term use of IDNA encoded sequences for the domainComponent relative distinguished names of the target directory tree. Since these sequences are URL-safe, they should require no additional encoding. However, if an internationalized equivalent of the domainComponent attribute were to be defined, then presumably those sequences would also have to be percent-hacked, similar to the way in which the entry names are described in the preceding paragraph. Applications should also be made aware of this through explicit text in the LDAP-WHOIS collection of specifications, if such a data-type should come to pass.

2.2.9 Client-Side Transformations

As can be surmised from the above discussions, there are several instances where an LDAP-WHOIS client will need to transform domain name input from UTF-8 to IDNA and vice-versa. However, there are also several areas where this may occur during output. In particular, a client may choose to automatically convert all IDNA sequences in attributes with a domain name syntax into UTF-8 (or vice-versa), with a user-selectable preference defining the output format.

Generally speaking, these kinds of conversions may be harmless as long as the client performs the proper (de)conversion if the data is subsequently used for input. Note that this may result from a variety of factors, including copy-and-paste operations, allowing the user to click on hyperlinked references (specifically reference attributes), and more.

Since these issues are mostly local in scope, they cannot be effectively dictated by a protocol/service specification. Furthermore, as long as the specifications adequately detail the appropriate syntaxes and any conversions which may be necessary, the risk of unintentional protocol pollution should be minimized. However, the specifications should highlight these risks, and should warn against wanton unidirectional conversion.

2.3 Other LDAP-WHOIS Issues

Apart from domain names, there are several other kinds of data which also have internationalization or localization considerations. These additional issues are described in this section.

2.3.1 Attribute Identifiers

Each LDAP attribute has a formal "name" which is typically used for local display purposes. Because these names act as simple identifiers, they can be localized through client-side conversions, and requires no additional support services.

2.3.2 Timestamps

Several of the attributes defined in LDAP-WHOIS provide timestamps. In the common case, these timestamps use the generalizedTime syntax, which provides a full date and time (including the timezone). Because these sequences are standardized parts of the LDAP service and are well-known, they may be converted into localized formats for display purposes without much difficulty. As such, localization of these identifiers is already possible, and requires no additional support services.

2.3.3 Postal Address Country-Codes

International postal rules generally suggest that postal addresses should be printed in the native language of the destination country, with the exception of the destination country code, which should be printed in the native language of the originator country. This model gives the originator postal office enough information to route the mail to the appropriate destination country, while ensuring that the destination postal office can make final delivery.

In order to facilitate this usage model, the country code information in LDAP-WHOIS must be tokenized into a form that is language-independent, thereby allowing each originator to convert the destination country code into their local language.

The current "country" attribute used for this purpose utilizes two-letter ISO 3166 country codes, which effectively act as tokenized representations of the destination country. In order to satisfy the postal requirements, it is recommended that client implementations convert these country codes into the localized equivalents of the associated country names, as appropriate.

2.3.4 Free-Text Attributes

Several of the attributes used in LDAP-WHOIS provide free-text data. These attributes are required to be provided with language tags so that the data can be presented and searched more efficiently, especially in multi-lingual environments.

Some of these attributes are inherited from other (pre-existing) schema definitions, while some of them are defined in LDAP-WHOIS. In general, almost all of them are capable of storing language tags through the standards-track mechanisms described in RFC 2596, which uses attribute descriptors to label and store alternative language representations of a common attribute value.

For example, the "organization" attribute could list a company's name in many different languages by using this syntax, with the client displaying one of these versions based on the current locale setting. This approach would also be useful with personal names, street addresses, disclaimer text, and most other kinds of free-text data.

The LDAP-WHOIS specification should be updated to specifically recommend this approach for most free-text attributes, and for certain attributes specifically (in particular, names, addresses and descriptive text).

Note that there are several attributes where this mechanism cannot be used, and those attributes must also be detailed. For example, several attributes make use of a syntax which allows free-text, although the attribute values themselves are structured data and are not free-text (examples of this include labeledURI and email addresses, both of which are described separately in this document).

2.3.5 URL Descriptions

As was discussed in section 2.2.8, many of the URLs used in LDAP-WHOIS make use of the labeledURI attribute, which is a two-part notation format consisting of a URI and a free-text string. Since the free-text portions of these strings use the directoryString syntax, they are already capable of supporting internationalized text. However, CRISP requires that all free-text data must be accompanied by a language identifier.

The free-text portion of this data is somewhat problematic towards this objective. Since this data is not separately packaged from the URL, any kind of tagging mechanism for the free-text data would have to make use of an in-band escape sequence, but one which did not interfere with other uses of the data. While it is possible to define such a sequence for use with LDAP-WHOIS in particular, this data is likely to be reused in other LDAP applications, and these tagging mechanisms may not be understood or properly generated within the context of those applications.

Note that there are mechanisms for providing language tags on entire LDAP attributes, but these tags are not appropriate for the labeledURI attribute as a whole, since the tags would also infer that the URL sub-part of the attribute was also governed by the selected language, which would not be appropriate (URLs are not free-text).

Due to these kinds of considerations, this specific issue requires further research before any recommendations can be made. In the meantime, the recommendation is for neither

the labeledURI attribute nor the subordinate free-text sequences to have no language tags associated with them.

3. IRIS Issues

The IRIS specifications effectively describe a vendor-neutral transfer format, rather than describing data-storage or application input and output behaviors. In general terms, this means that IRIS can be simplified substantially over LDAP-WHOIS, although a substantial amount of additional definitions are required to ensure interoperability within this simpler model.

3.1 XML normalizedString

Most of the domain name elements which are used throughout the IRIS schema make use of the well-known and standardized "normalizedString" data-type specified in the core XML Schema specifications [http://www.w3.org/TR/xmlschema-2/#normalizedString]. The normalizedString data-type is a sub-type of the "string" data-type with greater restrictions on the range of characters which may be used.

The normalizedString data-type appears to be sufficient for storing internationalized domain names as raw data. However, these IDNs should be normalized and validated before they are stored as XML data, and should be further validated upon being read. This behavior should be defined in the IRIS specifications to ensure interoperability.

3.2 IRIS Domain Name Instances

Domain names are used throughout IRIS, although only a handful of these instances are formally described. For example, the current specifications do not describe any presentation or matching behaviors. While this is in-line with its limited role as a specification for a transfer-format, this is not sufficient for the actual scope in which these domain names will eventually be used. In order to ensure interoperability, the IRIS specifications are likely to require significant extensions, particularly in terms of defining normalization routines for application clients. Most of the necessary modifications are described throughout the remainder of this document.

3.2.1 IRIS Elements

Almost all of the instances of a domain name within IRIS are simple elements, each of which use the normalizedString data-type as described in section 3.1.

These elements include:

- ?? <domainName>
- ?? <hostname>
- ?? <baseDomain>
- ?? <findDomainsByRegistrant> Query
- ?? <findDomainsByName> Query
- ?? <findDomainsByHost> Query

In all of these cases, the raw UTF-8 form of the IDN should be used. In order to ensure interoperability and data-integrity, the domain names should undergo a two-step conversion -- first converted into IDNA, and reconverted into UTF-8 -- before the domain names are stored as data in the XML elements.

It is strongly recommended that this process be detailed in subsequent versions of the IRIS specifications so as to prevent interoperability problems.

Note that the <hostname> element may prove to be more useful in its IDNA form, especially when these hostnames are passed to DNS applications (dig, host, etc.). In order to facilitate these usage scenarios, client applications should be encouraged to provide mechanisms which will allow the user to retrieve the IDNA-encoded representations of these elements.

3.2.2 Email Addresses

The <eMail> element provided for storing and passing email addresses uses the well-known XML "string" data-type (rather than the normalizedString data-type). This data-type is also capable of handling internationalized domain names in their raw form.

For the purpose of consistency, the domain name portion of email addresses which are stored in the <email> element should also make use of the raw UTF-8 encoding, rather than using the IDNA encoding. This process would ensure that international domain names are consistent and validated before they have been exchanged. As with the <hostname> element, client applications should also be encouraged to provide mechanisms which will allow the user to retrieve the IDNA-encoded representations of the <email> element as raw data.

The formatting rules for the local-part portion of email addresses in <email> elements should also allow for the use of raw UTF-8 sequences, even though the canonical formatting rules for email addresses do not currently allow these characters. However, it can be assumed that these characters will eventually be allowed, and this inevitability should be recognized in the IRIS specifications. Separate and apart from the underlying element syntax, applications should be encouraged to verify the local-part portion of any email addresses that they generate. In other words, an application can verify the local-part

syntax which has been entered locally and reject an apparently-illegal syntax, but that same application should be capable of displaying local-part syntaxes however they are provided by the remote system.

3.2.3 IRIS URLs

As with the LDAP-WHOIS specifications, IRIS makes frequent and extensive use of URLs to provide referrals and pointers, with a typical URL providing all of the information needed for a client to generate a fully-formed query.

There are two areas where an internationalized domain name may appear within the IRIS URL: the name of an entity which is (presumably) authoritative for the resource in question; and the domain name of the resource in question.

The following example shows these components in a referral:

iris://com/dreg/domain/example.com

where "com" is the entity that is authoritative for the data, and where "example.com" is the resource data in question.

Both of these elements have their own considerations.

The "authority" domain name will eventually be used to generate DNS lookups for SRV or A resource records, and therefore must use a domain name which can be resolved through DNS, which essentially requires IDNA to be used for this element. However, the procedure for converting an internationalized domain name into an IDNA string as part of the SRV processing can be defined to serve the same purpose. In order to promote consistency across the IRIS specifications, it is recommended that this latter approach be pursued, with the "authority" domain name being provided in its raw UTF-8 form, as with other instances.

The "entity-name" stores the domain name of the resource in question, and is somewhat more problematic. The current IRIS draft specification (-01) specifies that this sequence may use any of the unreserved characters from RFC 2396, and that it must be UTF-8 encoded with "application/x-www-form-urlencoded" as specified by RFC 1866. These instructions are somewhat complex, and provide no real benefit. In order to simplify processing, it is recommended that the "entity-name" element should also contain the raw UTF-8 form of the internationalized domain name.

In broad terms, it is recommended that the IRIS specifications should strictly define the "iris" URL protocol identifier as UTF-8 data, particularly when these URLs are exchanged as part of the IRIS service itself, and in other transfers which are capable of supporting UTF-8 data. Furthermore, although it does not appear that there are any immediate usage scenarios which will require the use of ASCII-restricted URLs, the use

of IRIS URLs within limited environments can be expected, so a standardized mechanism for encoding these URLs into ASCII should also be defined. Note that there is ongoing work towards defining and manipulating internationalized resource identifiers (IRIs, as opposed to URIs), including common mechanisms for conversions [see http://www.ietf.org/internet-drafts/draft-duerst-iri-03.txt]. This work should be taken into consideration as part of this specification.

3.2.4 SRV Processing

IRIS uses SRV resource records in much the same manner as LDAP-WHOIS, with the same basic requirements and ramifications. The Domain Name System itself currently requires the use of ASCII text, and this requirement also applies to the domain names associated with SRV resource records, including their owner domain names, and any domain names provided in the resource record data. As such, all SRV processing in IRIS must use the IDNA form of any internationalized domain names.

As was discussed in section 3.2.3, the process of converting an internationalized host identifier from an IRIS URL should be defined as part of the SRV-lookup process.

3.2.5 Input, Output, Storage and Comparison Formats

The IRIS specifications do not define any rules for user input or output formats, storage formats, or comparison formats. It is my recommendation that these areas be addressed in a future version of the specifications, and that the application end-points be made responsible for any conversion to and/or from IDNA as needed.

Specifically, domain names which are input as user query values should be converted into their raw UTF-8 representation, with these domain names being validated as part of this process. In order to ensure this consistency, all domain names should be converted to their encoded IDNA form and then decoded into UTF-8 before those domain names are used in the XML data-stream. This approach will ensure that servers are not presented with malformed data, thereby facilitating faster comparisons at the server, while also avoiding query mismatches.

The same model should also be used for output data, in that servers should only generate IDNs which have been validated (possibly on database-input, or possibly on-the-fly during response generation). This approach allows clients which use UTF-8 to handle the data without conversion, while also providing a "universal" format to clients which need to convert the data into a local charset or encoding format. Note that some kinds of data are excepted from this rule, and are encouraged to be provided in their IDNA forms in every case.

3.3 Other IRIS Issues

3.3.1 Attribute Identifiers

The XML elements in IRIS use textual strings as their object identifiers, with these strings being passed as part of the protocol data. However, these identifiers are not intended to serve as presentation identifiers, and are only used for data-tagging purposes. As such, these identifiers are readily capable of being treated as tokens, with clients using localized forms of these names for presentation purposes. As such, localization of these identifiers is already possible, and requires no additional support services.

3.3.2 Timestamps

Several of the XML elements defined in IRIS provide timestamps. In the common case, these timestamps use the standardized and well-known dateTime syntax, which provides a full date and time. Because these sequences are standardized and well-known, they may be converted into localized formats for display purposes without much difficulty. As such, localization of these identifiers is already possible, and requires no additional support services.

3.3.3 Postal Address Country-Codes

IRIS provides an XML element for country codes which is presumably a tokenized form of the ISO 3166 country codes, although this is not explicitly stated. It is recommended that this be stated if it is the case. Since ISO 3166 country codes allow for localization, no additional support services will be required.

3.3.4 Free-Text Attributes

IRIS already provides some language-tagging in certain areas, although a cursory examination of this feature appears to indicate that this is a very limited feature. For example, while it does provide for the tagging of free-text data such as copyright notices, it does not offer multiple languages of the same textual blocks. Furthermore, only a few elements are given this option, although many elements could (and should) benefit from these tags. Specifically, this capability needs to be added to contact names, organization names, postal addresses, and other unstructured textual data.

It is strongly recommended that all free-text elements be provided with alternative language representations, and that the behavioral rules for processing these sequences be fully described.

4. Summary Analysis and Recommendations

In broad terms, IRIS benefits from having a very limited scope, while LDAP-WHOIS suffers from having to share its database with other applications. Specifically, IRIS can be limited to specifying a vendor-neutral transfer format, while LDAP-WHOIS must share data with other LDAP applications and services and must be a good team player with those applications. IRIS can be unilateral in its specifications and this leads to some efficiencies, but LDAP-WHOIS has to be multilateral and this leads to some problems.

It is recommended that IRIS take advantage of its isolated nature to maximize on transfer efficiencies, specifically by embracing UTF-8 throughout the specification. This approach will allow end-point applications to provide immediate benefits to its users, while also ensuring that data in the protocol-stream is consistent.

Conversely, LDAP-WHOIS is driven by several long-term considerations which prevent this rapid adoption. LDAP data will absolutely get reused by other applications, and those applications absolutely need to be provided with the raw version of the internationalized domain names wherever possible, especially since many of those applications will be user-facing. Unfortunately, many of the attributes used in LDAP-WHOIS have issues which prevent embracing this approach throughout the specification (this is especially true in regards to the domainComponent syntax).

As a result of this uneven support, implementation of internationalized domain names throughout LDAP-WHOIS is significantly more difficult, requiring a case-by-case implementation. This approach is more expensive, more prone to error, and generally more troublesome. However, the long-term benefits are scalar to the cost, so this effort is not a pure resource sink. While it would be simpler to pursue an IDNA-only strategy throughout LDAP-WHOIS, the rewards for this approach would also be minimal and offering only a brief benefit to the community as a whole. As such, it is worth the effort to be a good team player, and to provide internationalized domain names in their raw form wherever possible, even though it means spotty implementation in the short term.