Network Working Group
Request for Comments: 5483
Category: Informational

L. Conroy
RMRL
K. Fujiwara
JPRS

March 2009

ENUM Implementation Issues and Experiences

Status of This Memo

This memo provides information for the Internet community. It does not specify an Internet standard of any kind. Distribution of this memo is unlimited.

Copyright Notice

Abstract

This document captures experiences in implementing systems based on the ENUM protocol and experiences of ENUM data that have been created by others. As such, it clarifies the ENUM and Dynamic Delegation Discovery System standards. Its aim is to help others by reporting both what is "out there" and potential pitfalls in interpreting the set of documents that specify the ENUM protocol. It does not revise the standards but is intended to provide technical input to future revisions of those documents.

Table of Contents

1. Introduction

1.1. Document Goal

The goal of this document is to clarify the ENUM and Dynamic Delegation Discovery System (DDDS) standards. It does not itself revise ENUM or DDDS standards but is intended to provide technical input to future revisions of those documents. It also serves to advise implementers on the pitfalls that they may find. It highlights areas where ENUM implementations have differed over interpretation of the standards documents or have outright failed to implement some features as specified.

As well as providing clarifications to standards text, this document also mentions potential choices that can be made, in an attempt to help foster interworking between components that use this protocol. The reader is reminded that others may make different choices.

The core specifications for the E.164 Number Mapping (ENUM) protocol [RFC3761] and the Dynamic Delegation Discovery System (DDDS) [RFC3403] [RFC3401] [RFC3402] [RFC3404] [RFC3405] are defined elsewhere. Unfortunately, this document cannot provide an overview of the specifications, so the reader is assumed to have read and understood the complete set of ENUM normative documents.

The Domain Name System (DNS) is ENUM's database. ENUM uses the NAPTR (Naming Authority Pointer) resource record type to store its DDDS rules into DNS domains. ENUM relies on DNS services. Thus, it is also important for ENUM implementers to carry out a thorough analysis of all of the existing DNS standard documents to understand what services are provided to ENUM and what load ENUM provisioning and queries will place on the DNS.

A great deal of the rationale for making the choices listed in this document is available to those who explore the standards. The trick of course is in understanding those standards and the subtle implications that are involved in some of their features. In almost all cases, the choices presented here are merely selections from values that are permissible within the standards.

1.2. Terminology

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in [RFC2119].

2. Character Sets and ENUM

2.1. Character Sets - Non-ASCII Considered Harmful

[RFC3403] and [RFC3761] specify respectively that NAPTR resource records and ENUM support Unicode using the UTF-8 encoding defined in [RFC3629]. This raises an issue when implementations use "single byte" string-processing routines. If there are multi-byte characters within an ENUM NAPTR, incorrect processing may well result from these UTF-8-unaware systems.

The UTF-8 encoding has a US-ASCII equivalent range, so that all characters in US-ASCII [ASCII] from 0x00 to 0x7F hexadecimal have an identity map to the UTF-8 encoding; the encodings are the same. In UTF-8, characters with Unicode code points above this range will be encoded using more than one byte, all of which will be in the range 0x80 to 0xFF hexadecimal. Thus, it is important to consider the different fields of a NAPTR and whether or not multi-byte characters can or should appear in them.

In addition, characters in the non-printable portion of US-ASCII (0x00 to 0x1F hexadecimal, plus 0x7F hexadecimal) are "difficult". Although NAPTRs are processed by machine, they may sometimes need to be written in a human-readable form. Specifically, if NAPTR content is shown to an end user so that he or she may choose, it is imperative that the content is human-readable. Thus, it is unwise to use non-printable characters even if they lie within the US-ASCII range; the ENUM client may have good reason to reject NAPTRs that include these characters as they cannot readily be presented to an end user.

There are two numeric fields in a NAPTR: the ORDER and PREFERENCE/PRIORITY fields. As these contain binary values, no risk is involved because string processing should not be applied to them. The string-based fields are the Flags, Services, and Regexp fields. The Replacement field holds an uncompressed domain name, encoded according to the standard DNS mechanism [RFC1034][RFC1035]. The Internationalised Domain Name (IDN) can be supported (as specified in [RFC3490], [RFC3491], and [RFC3492]). Any such IDN MUST be further encoded using Punycode [RFC3492]. As the Replacement field holds a domain name that is not subject to replacement or modification (other than Punycode processing), it is not of concern here.

Taking the string fields in turn, the Flags field contains characters that indicate the disposition of the NAPTR. This may be empty, in which case the NAPTR is "non-terminal", or it may include a flag

character as specified in [RFC3761]. These characters all fall into the printable US-ASCII equivalent range, so multi-byte characters cannot occur.

The Services field includes the DDDS Application identifier ("E2U") used for ENUM, a set of Enumservice identifiers, any of which may embed the ':' separator character, together with the '+' character used to separate Enumservices from one another and from this DDDS Application identifier. In Section 2.4.2 of [RFC3761], Enumservice identifier tokens are specified as 1*32 ALPHA/DIGIT, so there is no possibility of non-ASCII characters in the Services field.

2.1.1. Non-ASCII in the Regular Expression Field

The Regexp field is more complex. It forms a sed-like substitution expression, defined in [RFC3402], and consists of two sub-fields:

- a POSIX Extended Regular Expression (ERE) sub-field [IEEE.1003-2.1992]

- a replacement (Repl) sub-field [RFC3402].

Additionally, [RFC3402] specifies that a flag character may be appended, but the only flag currently defined there (the 'i' case-insensitivity flag) is not appropriate for ENUM -- see Section 2.2.

The ERE sub-field matches against the "Application Unique String"; for ENUM, this is defined in [RFC3761] to consist of digit characters, with an initial '+' character. It is similar to a global-number-digits production of a tel: URI, as specified in [RFC3966], but with visual-separators removed. In short, it is a telephone number (see [E.164]) in restricted format. All of these characters fall into the US-ASCII equivalent range of UTF-8 encoding, as do the characters significant to the ERE processing.

Strictly, the ERE might include other characters. The ERE could include choice elements matching against different items, some of which might not be an ENUM Application Unique String. Those alternative matching elements might conceivably include non-ASCII characters. As an operational issue, it is not reasonable to include such constructs, as ENUM NAPTRs match against telephone numbers.

In the normal situation in which E2U NAPTRs are provisioned in ENUM domains, there will be no multi-byte characters within this sub-field, as the ERE will be intended to match against telephone numbers. ENUM clients must be able to handle NAPTRs that do contain such multi-byte characters (as the standard does not preclude them), but there is no operational reason for these ever being provisioned

in ENUM domains. If NAPTRs provisioned in ENUM domains are encountered containing such multi-byte characters, these could reasonably be discarded.

The Repl sub-field can include a mixture of explicit text used to construct a URI and characters significant to the substitution expression, as defined in [RFC3403]. Whilst the latter set all fall into the US-ASCII equivalent range of UTF-8 encoding, this might not be the case for all conceivable text used to construct a URI. Presence of multi-byte characters could complicate URI generation and processing routines.

URI generic syntax is defined in [RFC3986] as a sequence of characters chosen from a limited subset of the repertoire of US-ASCII characters. The current URIs use the standard URI character escaping rules specified in the URI generic syntax, and so any multi-byte character will be pre-processed; they will not occur in the explicit text used to construct a URI within the Repl sub-field.

2.1.1.1. Impact of Future Support for IRIs

As currently specified, ENUM only permits URIs to be generated in the Regexp field. However, even if this were to be extended in future revisions of the ENUM specification to allow the use of Internationalised Resource Identifiers (IRIs), defined in [RFC3987], further support for non-ASCII characters may be avoided. IRIs are defined as extending the syntax of URIs, and RFC 3987 specifies a mapping from IRIs to URIs. IRI syntax allows characters with multi-byte UTF-8 encoding.

Given that this is the only place within an ENUM NAPTR where such multi-byte encodings might reasonably be found, a simple solution is to use the mapping method specified in Section 3.1 of [RFC3987] to convert any IRI into its equivalent URI.

This process consists of two elements; the domain part of an IRI MUST be processed using Punycode if it has a non-ASCII domain name, and the remainder MUST be processed using the extended escaping rules specified in [RFC3987] if it contains characters outside the normal URI repertoire. Using this process, there will be no non-ASCII characters in any part of any URI, even if it has been converted from an IRI that contains such characters.

2.1.2. Non-ASCII Support - Conclusions

From the analysis just given, the only place within an ENUM NAPTR where non-ASCII characters might be found is the Regexp field. It is possible to remove any requirement to process characters outside the

US-ASCII equivalent range by adding very few operational restrictions. There is no obvious benefit in providing characters outside this range. Handling multi-byte characters complicates development and operation of client programs, and many existing programs do not include such support.

As the gain from permitting characters outside the US-ASCII equivalent range is unclear, and the costs of multi-byte character processing are very clear, ENUM NAPTRs SHOULD NOT include characters outside the printable US-ASCII equivalent range.

2.2. Case Sensitivity

The only place where NAPTR field content is case sensitive is in any static text in the Repl sub-field of the Regexp field. Everywhere else, case-insensitive processing can be used.

The case-insensitivity flag ('i') could be added at the end of the Regexp field. However, in ENUM, the ERE sub-field operates on a string defined as the '+' character, followed by a sequence of digit characters. This flag is redundant for E2U NAPTRs, as it does not act on the Repl sub-field contents.

Thus, the case-sensitivity flag is inappropriate for ENUM, and SHOULD NOT be provisioned into E2U NAPTRs.

2.3. Regexp Field Delimiter

It is not possible to select a delimiter character that cannot appear in one of the sub-fields. The '!' character is used as a delimiter in all of the examples in [RFC3403] and in [RFC3761]. It is the only character seen in existing zones, and a number of different client implementations are still "hardwired" to expect this character as a delimiter.

The '!' character will not normally appear in the ERE sub-field. It may appear in the content of some URIs, as it is a valid character (e.g., in http URLs). If it is present in the Regexp field, then that instance MUST be escaped using the standard technique proposed in Section 3.2 of [RFC3402]: a backslash character (U+005C) should be inserted before it in the string. Otherwise, a client may attempt to process this as a standard delimiter and interpret the Regexp field contents differently from the system that provisioned it.

2.4. Regexp Meta-Character Issue

In ENUM, the ERE sub-field may include a literal character '+', as the Application Unique String on which it operates includes this. However, if it is present, then '+' MUST be escaped using a single backslash character (to produce the sub-string U+005C U+002B), as '+' is a meta-character in POSIX Extended Regular Expression syntax.

Not escaping the '+' character produces an invalid ERE, but is a common mistake. Even standards have given incorrect examples; the obsolete [RFC2916] (Section 3.4.3, example 3) has this problem.

For example, the following NAPTR example is incorrect:

```
* IN NAPTR 100 10 "u" "E2U+sip" "!^+4655(.*)$!sip:\\1@example.net!" .
```

A correct way to write this example is:

```
* IN NAPTR 100 10 "u"
    "E2U+sip" "!^\\+4655(.*)$!sip:\\1@example.net!" .
```

Note that when a NAPTR resource record is shown in DNS master file syntax (as in this example above), the backslash itself must be escaped using a second backslash. The DNS on-the-wire packet will have only a single backslash.

3. Unsupported NAPTRs

An ENUM client MAY discard a NAPTR received in response to an ENUM query because:

- the NAPTR is syntactically or semantically incorrect,
- the NAPTR has a different (non-empty) DDDS Application identifier from the 'E2U' used in ENUM,
- the NAPTR's ERE does not match the Application Unique String for this ENUM query,
- the ENUM client does not recognise any Enumservice held in this NAPTR, or
- this NAPTR (only) contains an Enumservice that is unsupported.

These conditions SHOULD NOT cause the whole ENUM query to terminate, and processing SHOULD continue with the next NAPTR in the returned Resource Record Set (RRSet).

When an ENUM client encounters a compound NAPTR (i.e., one containing more than one Enumservice -- see also Section 4.4.1) and cannot process or cannot recognise one of the Enumservices within it, that ENUM client SHOULD ignore this Enumservice and continue with the next Enumservice within this NAPTR's Services field, discarding the NAPTR only if it cannot handle any of the Enumservices contained. These conditions SHOULD NOT be considered errors.

ENUM uses regular-expression processing when generating URIs from the Regexp field of "terminal" NAPTRs. Just as with all uses of regular expressions, there is a potential for buffer overrun when generating this output. There may be repeated back-reference patterns in a NAPTR's Repl sub-field, and the output these generate may consume a considerable amount of buffer space.

Even if an ENUM client would normally encounter only NAPTRs with short URIs, it may also receive NAPTRs with repeated back-reference patterns in their Repl sub-fields that could generate strings longer than the client's buffer. Such NAPTRs may have been misconfigured accidentally or by design. The client MUST NOT fail in this case. It SHOULD NOT discard the entire ENUM query, but instead just discard the NAPTR that would otherwise have caused this overrun.

If a problem is detected when processing an ENUM query across multiple domains (by following non-terminal NAPTR references), then the ENUM query SHOULD NOT be abandoned, but instead processing SHOULD continue at the next NAPTR after the non-terminal NAPTR that referred to the domain in which the problem would have occurred. See Section 5.2.2 for more details.

3.1. Non-Compliant Client Behaviour

Through monitoring current ENUM clients, a number of non-compliant behaviours have been detected. These behaviours are incorrect, but may be encountered in still-operational client implementations.

ENUM clients have been known to discard NAPTRs in which the Services field holds more than one Enumservice.

ENUM clients have also been known to discard NAPTRs with a "non-greedy" ERE sub-field expression (i.e., EREs that are dissimilar to "^.*$").

ENUM clients have been known to discard NAPTRs that do not use '!' as their Regexp delimiter character.

ENUM clients have been known to discard NAPTRs in which the delimiter is NOT the last character in the Regexp field.

ENUM clients have been known to discard NAPTRs with an empty Flags field (i.e., "non-terminal" NAPTRs).

ENUM clients have been known to ignore the ORDER field value entirely, sorting the NAPTRs in an RRSet based solely on the PREFERENCE/PRIORITY field values.

Finally, many ENUM clients have been known to discard a NAPTR where they have local knowledge that the URI that would be generated by processing the NAPTR is unusable. This behaviour is, strictly speaking, non-compliant, but might be considered reasonable (see Section 4.1).

4. ENUM NAPTR Processing

ENUM is a DDDS Application, and the way in which NAPTRs in an RRSet are processed reflects this. The details are described in Section 3.3 of [RFC3402]. The client is expected to sort the records it receives into a sequence and then process those records in that sequence. The sequence reflects the ORDER and PREFERENCE/PRIORITY field values in each of the NAPTRs.

The ORDER field value is the major, or most significant, sort term and the PREFERENCE/PRIORITY field value is the minor, or least significant, sort term. The combination of ORDER and PREFERENCE/PRIORITY field values indicates the sequence chosen by the publisher of this data, and NAPTRs will be considered in this sequence.

Once the NAPTRs are sorted into sequence, further processing is done to determine if each of the NAPTRs is appropriate for this ENUM evaluation. This involves looking at the Flags field. If the Flags field is empty, this is a "non-terminal" NAPTR and is processed as described in Section 5.

If the "u" Flag is present (and so the NAPTR is a "terminal" rule that generates a URI), the Services field is checked to ensure that this NAPTR is intended for ENUM (i.e., that this NAPTR includes the "E2U" DDDS Application identifier in the Services field). The ERE in the Regexp field is checked and must match the Application Unique String (AUS) for this ENUM evaluation (the queried telephone number). Unless each of these checks succeeds, the NAPTR is discarded and the next in sequence is processed.

During this processing, clients will also consider the Enumservices within the Services field. Enumservices indicate the kind of interaction that can be achieved through use of the URI this NAPTR generates. If there is local knowledge that a NAPTR includes only an Enumservice that is either not supported or not recognised, then this

NAPTR can be discarded and the next in sequence will be processed. Thus, for a system that has support only for SIP interactions, if it receives an RRSet in which the "best" NAPTR indicates the H323 Enumservice, then that client could reasonably discard that NAPTR and go on to the next in sequence.

4.1. Common Non-Compliant ENUM Processing

The processing of ORDER and PREFERENCE/PRIORITY fields has been a significant source of confusion, and many ENUM clients do not implement the processing exactly as specified.

In particular, many ENUM clients use local prior knowledge about URIs during ENUM processing. If a client has prior knowledge that a particular URI will not result in an acceptable outcome, it might discard that NAPTR and consider the next one in the sequence. Examples of such local prior knowledge include: the URI does not resolve, authentication has been recently rejected, or user policies mark a particular URI as unacceptable (the URI could be a "premium rate" telephone number that would be charged at an unacceptable rate).

Strictly speaking, this behaviour is non-compliant if the next NAPTR record has a different ORDER value. The ENUM algorithm (Section 3.3 of [RFC3402] and Section 4.1 of [RFC3403]) states that once a match has been found for the Application Unique String (AUS), and the service description satisfies the client's requirements, NAPTR records with larger ORDER values must not be considered (but other NAPTR records with the same ORDER value can still be considered).

However, embedding local knowledge about the URI within the ENUM evaluation process is almost universal in systems employing ENUM. Also, since the difference between ORDER and PRIORITY/PREFERENCE has been unclear, NAPTR records have been provisioned in ways that would make strictly compliant systems unusable in practice. Given that such systems are intended to provide communications, this non-compliant, "embedded decision" behaviour is understandable.

It is proposed that when the ENUM specification is updated, processing of ORDER and PRIORITY/PREFERENCE should be updated based on implementation and deployment experiences described in this document.

4.1.1. Example

The example in this section is intended to further understanding about the difference between what [RFC3402] and [RFC3403] specify and what existing ENUM clients do.

WARNING: The NAPTR records shown in this section are intended to illustrate somewhat unclear corner cases, and are not intended as good examples of how to do ENUM provisioning.

Consider the following RRset, which maps numbers in the UK drama range to one server, and all other numbers to a second server:

```
* 3600 IN NAPTR 1 1 "u" "e2u+sip"
    "!^(\\+441632960.*)$!sips:\\1@atlanta.example.com!" .
* 3600 IN NAPTR 2 1 "u" "e2u+sip"
    "!^(.*)$!sip:\\1@biloxi.example.com!" .
```

According to the processing specified in [RFC3402] and [RFC3403], the ENUM client is never intended to consider the second rule for e.g., AUS "+441632960123", even if it does not support "sips" URIs, or the atlanta.example.com server cannot be reached, or the user indicates he or she doesn't wish to contact atlanta.example.com. However, existing ENUM implementations are known to do this, and as described above, it can be useful if the alternative is failing to communicate at all.

To prevent a client from considering the second rule for the UK drama range, the example could be rewritten to have more predictable behaviour as follows:

```
* 3600 IN NAPTR 1 1 "u" "e2u+sip"
    "!^(\\+441632960.*)$!sips:\\1@atlanta.example.com!" .
* 3600 IN NAPTR 2 1 "u" "e2u+sip"
    "!^(\\+[^4].*|\\+4[^4].*|\\+44[^1].*|\\+441[^6].*|\\+4416[^3].*|
    \\+44163[^2].*|\\+441632[^9].*|\\+4416329[^6].*|
    \\+44163296[^0].*)$!sip:\\1@biloxi.example.com!" .
```

4.2. Order/Priority Values - Processing Sequence

[RFC3761] and [RFC3403] state that the ENUM client MUST sort the NAPTRs using the ORDER field value ("lowest value is first") and SHOULD order the NAPTRs using the PREFERENCE/PRIORITY field value as the minor sort term (again, lowest value first). The NAPTRs in the sorted list must be processed in order. Subsequent NAPTRs with worse ORDER values must only be dealt with once the current ones with a better ORDER value have been processed.

However, as described in the introduction to this section, this stated behaviour is a simplification. Once sorted into a sequence reflecting ORDER and PREFERENCE/PRIORITY values, other fields are also considered during evaluation of retrieved NAPTRs; local knowledge may play a factor in the decision process, once a NAPTR has reached that point in the sequence at which it is considered.

ENUM clients may also include the end user "in the decision loop", offering the end user the choice from a list of possible NAPTRs. Conceptually this choice is embedded within step 4 of the DDDS algorithm (as described in Section 3.3 of [RFC3402]). Given that the ORDER field value is the major sort term, one would expect a conforming ENUM client to present only those NAPTRs with the currently "best" ORDER field value as choices. When/if all the presented options had been rejected, then the ENUM client might offer those with the "next best" ORDER field value, and so on. As this may be confusing for the end user, some clients simply offer all of the available NAPTRs as options to the end user for his or her selection at once, in the sequence defined by the ORDER and PREFERENCE/PRIORITY fields.

In summary, ENUM clients will take into account the Services field value, the Flags field, and the Regexp ERE sub-field, along with the ORDER and PREFERENCE/PRIORITY field values, and may consider local policies or available local knowledge.

The Registrant and the ENUM zone provisioning system he or she uses must be aware of this and SHOULD NOT rely on ENUM clients solely taking account of the value of the ORDER and the PREFERENCE/PRIORITY fields alone.

Specifically, it is unsafe to assume that an ENUM client will not consider another NAPTR if there is one with a better ORDER value. The instructions in Section 4.1 and Section 8 of [RFC3403] may or may not be followed strictly by different ENUM clients for perfectly justifiable reasons.

Where the ENUM client presents a list of possible URLs to the end user for his or her choice, it MUST do so in the sequence defined by the ORDER and PREFERENCE/PRIORITY values specified by the Registrant.

However, a Registrant SHOULD place into his or her zone only contacts that he or she is willing to support; even those with the worst ORDER and PREFERENCE/PRIORITY values MAY be selected by an end user.

4.3. Use of Order and Preference Fields

NAPTRs in ENUM zones that hold incorrect ORDER values can cause major problems. [RFC3403] highlights that having both ORDER and PREFERENCE/PRIORITY fields is a historical artifact of the NAPTR resource record type. It is reasonable to have a common default value for the ORDER field, relying on the PREFERENCE/PRIORITY field to indicate the preferred sort.

We have noticed a number of ENUM domains with NAPTRs that have identical PREFERENCE/PRIORITY field values and different ORDER values. This may be the result of an ENUM zone provisioning system "bug" or a misunderstanding over the uses of the two fields, or simply a difference of interpretation of the standards.

To clarify, the ORDER field value is the major sort term, and the PREFERENCE/PRIORITY field value is the minor sort term. Thus, one should expect to have a set of NAPTRs in a zone with identical ORDER field values and different PREFERENCE/PRIORITY field values; not the other way around.

To avoid these common interoperability issues, it is recommended that ENUM NAPTRs SHOULD hold a default value in their ORDER field.

4.4. NAPTRs with Identical ORDER/PRIORITY Values

From experience, it has been learned that there are zones that hold discrete NAPTRs with identical ORDER and identical PREFERENCE/PRIORITY field values. This will lead to indeterminate client behaviour and so SHOULD NOT normally occur.

Such a condition indicates that these NAPTRs are truly identical in priority and that there is no preference between the services these NAPTRs offer. Implementers SHOULD NOT assume that the DNS will deliver NAPTRs within an RRSet in a particular sequence.

Multiple NAPTRs with identical ORDER and identical PREFERENCE/PRIORITY field values SHOULD NOT be provisioned into an RRSet unless the intent is that these NAPTRs are truly identical in priority and there is no preference between them.

Some ENUM client implementations have considered this case to be an error and have rejected such duplicates entirely. Others have attempted to further randomise the order in which such duplicates are processed. Thus, use of such duplicate NAPTRs is unwise, as client implementations exist that will behave in different ways.

4.4.1. Compound NAPTRs and Implicit ORDER/REFERENCE Values

With [RFC3761], it is possible to have more than one Enumservice associated with a single NAPTR. These Enumservices share the same Regexp field and so generate the same URI. Such a "compound" NAPTR could well be used to indicate a mobile phone that supports both "voice:tel" and "sms:tel" Enumservices. The Services field in that case would be "E2U+voice:tel+sms:tel".

A compound NAPTR can be treated as a set of NAPTRs that each hold a single Enumservice. These reconstructed NAPTRs share the same ORDER and PREFERENCE/PRIORITY field values but should be treated as if each had a logically different priority. In this case, the reconstructed NAPTR holding the leftmost Enumservice within the compound NAPTR has the best priority, and the reconstructed NAPTR holding the rightmost Enumservice has the worst priority in this set.

To avoid indeterminate behaviour, it is recommended that ENUM clients SHOULD process the Enumservices within a compound NAPTR in a left-to-right sequence. ENUM provisioning systems SHOULD assume that such a processing order will be used and provision the Enumservices within a compound NAPTR accordingly.

4.5. Processing Order Value across Domains

Using a different ORDER field value in different domains is unimportant for most queries. However, DDDS includes a mechanism for continuing a search for NAPTRs in another domain by including a reference to that other domain in a "non-terminal" NAPTR. The treatment of non-terminal NAPTRs is covered in the next section. If they are supported, then the way that ORDER and PREFERENCE/PRIORITY field values are processed is affected.

Two main questions remain from the specifications of DDDS and [RFC3761]:

- If there is a different (lower) ORDER field value in a domain referred to by a non-terminal NAPTR, then does this mean that the ENUM client discards any remaining NAPTRs in the referring RRSet?

- Conversely, if the domain referred to by a non-terminal NAPTR contains entries that only have a higher ORDER field value, then does the ENUM client ignore those NAPTRs in the referenced domain?

Whilst one interpretation of [RFC3761] is that the answer to both questions is "yes", this is not the way that those examples of non-terminal NAPTRs that do exist (and those ENUM clients that support them) seem to be designed.

In keeping with the interpretation made so far, ENUM implementations MUST consider the ORDER and PREFERENCE/PRIORITY values only within the context of the domain currently being processed in an ENUM query. These values MUST be discarded when processing other RRSets in the query.

5. Non-Terminal NAPTR Processing

5.1. Non-Terminal NAPTRs - Necessity

Consider an ENUM RRSet that contains a non-terminal NAPTR record. This non-terminal NAPTR holds, as its target, another domain that has a set of NAPTRs. In effect, this is similar to the non-terminal NAPTR being replaced by the NAPTRs contained in the domain to which it points.

It is possible to have a non-terminal NAPTR in a domain that is, itself, pointed to by another non-terminal NAPTR. Thus, a set of domains forms a "chain", and the list of NAPTRs to be considered is the set of all NAPTRs contained in all of the domains in that chain.

For an ENUM management system to support non-terminal NAPTRs, it is necessary for it to be able to analyse, validate, and (where needed) correct not only the NAPTRs in its current ENUM domain but also those referenced by non-terminal NAPTRs in other domains. If the domains pointed to have non-terminal NAPTRs of their own, the management system will have to check each of the referenced domains in turn, as their contents form part of the result of a query on the "main" ENUM domain. The domain content in the referenced domains may well not be under the control of the ENUM management system, and so it may not be possible to correct any errors in those RRSets. This is both complex and prone to error in the management system design, and any reported errors in validation may well be non-intuitive for users.

For an ENUM client, supporting non-terminal NAPTRs can also be difficult. Processing non-terminal NAPTRs causes a set of sequential DNS queries that can take an indeterminate time, and requires extra resources and complexity to handle fault conditions like non-terminal loops. The indeterminacy of response time makes ENUM-supported Telephony Applications difficult (such as in an "ENUM-aware" Private Branch Exchange (PBX)), whilst the added complexity and resources needed makes support problematic in embedded devices like "ENUM-aware" mobile phones.

Given that, in principle, a non-terminal NAPTR can be replaced by the NAPTRs in the domain to which it points, support of non-terminal NAPTRs is not needed and non-terminal NAPTRs may not be useful. Furthermore, some existing ENUM clients do not support non-terminal NAPTRs and ignore them if received.

To avoid interoperability problems, some kind of acceptable advice is needed on non-terminal NAPTRs. As current support is limited, non-terminal NAPTRs SHOULD NOT be used in ENUM unless it is clear that all of the ENUM clients this environment supports can process these.

5.2. Non-Terminal NAPTRs - Considerations

The following specific issues need to be considered if non-terminal NAPTRs are to be supported in a particular environment. These issues are gleaned from experience and indicate the kinds of conditions that should be considered before support for non-terminal NAPTRs is contemplated. Note that these issues are in addition to the point just mentioned on ENUM provisioning or management system complexity and the potential for that management system to have no control over the zone contents to which non-terminal NAPTRs in its managed zones refer.

5.2.1. Non-Terminal NAPTRs - General

As mentioned earlier, a non-terminal NAPTR in one RRSet refers to the NAPTRs contained in another domain. The NAPTRs in the domain referred to by the non-terminal NAPTR may have a different ORDER value from that in the referring non-terminal NAPTR. See Section 4.5 for details.

5.2.2. Non-Terminal NAPTRs - Loop Detection and Response

Where a chain of non-terminal NAPTRs refers back to a domain already traversed in the current query, a "non-terminal" or referential loop is implied. An implementation MAY treat a chain of more than 5 domains traversed during a single ENUM query as an indication that a self-referential loop has been entered.

There are many techniques that can be used to detect such a loop, but the simple approach of counting the number of domains queried in the current ENUM query suffices.

Where a loop has been detected, processing SHOULD continue at the next NAPTR in the referring domain (i.e., after the non-terminal NAPTR that included the reference that triggered the loop detection).

5.2.3. Field Content in Non-Terminal NAPTRs

The set of specifications defining DDDS and its applications are complex and multi-layered. This reflects the flexibility that the system provides but does mean that some of the specifications need clarification as to their interpretation, particularly where non-terminal rules are concerned.

5.2.3.1. Flags Field Content with Non-Terminal NAPTRs

Section 2.4.1 of [RFC3761] states that the only flag character valid for use with the "E2U" DDDS Application is 'u'. The flag 'u' is defined (in Section 4.3 of [RFC3404]) thus: 'The "u" flag means that the output of the Rule is a URI'.

Section 2.4.1 of [RFC3761] also states that an empty Flags field indicates a non-terminal NAPTR. This is also the case for other DDDS Application specifications, such as that specified in [RFC3404]. One could well argue that this is a feature potentially common to all DDDS Applications, and so might have been specified in [RFC3402] or [RFC3403].

The Flags field will be empty in non-terminal NAPTRs encountered in ENUM processing. ENUM does not have any other way to indicate a non-terminal NAPTR.

5.2.3.2. Services Field Content with Non-Terminal NAPTRs

Furthermore, [RFC3761] states that any Enumservice Specification requires definition of the URI that is the expected output of this Enumservice. This means that, at present, there is no way to specify an Enumservice that is non-terminal; such a non-terminal NAPTR has, by definition, no URI as its expected output, instead returning a key (DNS domain name) that is to be used in the "next round" of DDDS processing.

This in turn means that a non-terminal NAPTR cannot hold a valid (non-empty) Services field when used in ENUM. Section 2.4.2 of [RFC3761] specifies the syntax for this field content and requires at least one element of type <servicespec> (i.e., at least one Enumservice identifier). Given that there cannot be a non-terminal Enumservice (and so no such Registered Enumservice identifier), this syntax cannot be met with a non-terminal NAPTR; there are no non-terminal Enumservices to put into this field.

A reasonable interpretation of the specifications is that for a non-terminal NAPTR, the Services field must also be empty. This appears to be the approach taken by those clients that do either process non-terminal NAPTRs or check the validity of the fields.

It is expected that future revisions of the ENUM standard will clarify this text, making this interpretation plain. This was the intent of the current standard, and the intent will be made explicit in its revision.

In keeping with existing implementations, in a non-terminal NAPTR encountered in an ENUM query, the Services field SHOULD be empty, and clients SHOULD ignore any content it contains.

Of course, such non-terminal NAPTRs with an empty Services field are not specific to any DDDS Application. Thus, other means must be used to ensure a non-terminal NAPTR that is intended only for a particular DDDS Application cannot be encountered during a lookup for another DDDS Application (for example, by ensuring that the same domain is not used to host NAPTRs for more than one such DDDS Application).

5.2.3.3. Regular Expression and Replacement Field Content with Non-Terminal NAPTRs

The descriptive text in Section 4.1 of [RFC3403] is intended to explain how the fields are to be used in a NAPTR. However, the descriptions associated with the Regexp and Replacement elements have led to some confusion over which of these should be considered when dealing with non-terminal NAPTRs.

[RFC3403] is specific; these two elements are mutually exclusive. This means that if the Regexp element is not empty, then the Replacement element must be empty, and vice versa. However, [RFC3403] does not specify which is used with terminal and non-terminal rules.

The descriptive text of Section 4.1 of [RFC3403] for the NAPTR Replacement element shows that this element holds an uncompressed domain name. Thus, it is clear that this element cannot be used to deliver the terminal string for any DDDS Application that does not have a domain name as its intended terminal output.

However, the first paragraph of descriptive text for the NAPTR Regexp element has led to some confusion. It appears that the Regexp element is to be used to find "the next domain name to lookup". This might be interpreted as meaning that a client program processing the DDDS Application could need to examine each non-terminal NAPTR to decide whether the Regexp element or instead the Replacement element should be used to construct the key (a domain name) to be used next in non-terminal rule processing.

Given that a NAPTR holding a terminal rule (a "terminal NAPTR") must use the Substitution expression field to generate the expected output of that DDDS Application, the Regexp element is also used in such rules. Indeed, unless that DDDS Application has a domain name as its terminal output, the Regexp element is the only possibility.

Thus, from the descriptive text of this section, a Replacement element can be used only in NAPTRs holding a non-terminal rule (a "non-terminal NAPTR") unless that DDDS Application has a domain name as its terminal output, whilst the alternative Regexp element may be used either to generate a domain name as the next key to be used in the non-terminal case or to generate the output of the DDDS Application.

Note that each DDDS Application is free to specify the set of flags to be used with that application. This includes specifying whether a particular flag is associated with a terminal or non-terminal rule, and also includes specifying the interpretation of an empty Flags field (i.e., whether this is to be interpreted as a terminal or non-terminal rule, and if it is terminal, then what is the expected output). ENUM (as specified in Section 2.4.1 of [RFC3761]) uses only the 'u' flag, with an empty Flags field indicating a non-terminal NAPTR.

The general case in which a client program must check which of the two elements to use in non-terminal NAPTR processing complicates implementation, and this interpretation has NOT been made in current ENUM implementations. It would be useful to define exactly when a client program can expect to process the Regexp element and when to expect to process the Replacement element, if only to improve robustness. Generating an ENUM domain name from the Regexp field is difficult at best and impossible for the general case of a variable-length telephone number, or one that has more than 9 digits. Thus, it is proposed that when the ENUM specification is updated, this option is deprecated, and using the Regexp field for non-terminal ENUM NAPTRs is prohibited.

In keeping with current implementations, the target domain of a non-terminal ENUM NAPTR MUST be placed in the (non-empty) Replacement field. This field MUST be interpreted as holding the domain name that forms the next key output from this non-terminal rule. Conversely, the Regexp field MUST be empty in a non-terminal NAPTR encountered in ENUM processing, and ENUM clients MUST ignore its content.

6. Backwards Compatibility

6.1. Services Field Syntax

[RFC3761] is the current standard for the syntax for NAPTRs supporting the ENUM DDDS Application. This obsoletes the original specification that was given in [RFC2916]. RFC 3761 made a change to the syntax of the Services field of the NAPTR that reflects a refinement of the concept of ENUM processing.

As defined in [RFC3403], there is now a single identifier that indicates the DDDS Application. In the obsolete specification [RFC2915], there were zero or more "Resolution Service" identifiers (the equivalent of the DDDS Application). The same identifier string for the DDDS identifier or the Resolution Service is defined in both the [RFC3761] and [RFC2916] specifications: "E2U".

Also, [RFC3761] defines at least one but potentially several Enumservice sub-fields; in the obsolete specification, only one "protocol" sub-field was allowed.

In many ways, the most important change for implementations is that the order of the sub-fields has been reversed. [RFC3761] specifies that the DDDS Application identifier is the leftmost sub-field, followed by one or more Enumservice sub-fields, each separated by the '+' character delimiter. [RFC2916] specified that the protocol sub-field was the leftmost, followed by the '+' delimiter, in turn followed by the "E2U" resolution service tag.

[RFC2915] and [RFC2916] have been obsoleted by [RFC3401] - [RFC3404] and by [RFC3761]. However, [RFC3824] suggests that ENUM clients should be prepared to accept NAPTRs with the obsolete syntax. Thus, an ENUM client implementation may have to deal with both forms. This need not be difficult. For example, an implementation could process the Services field into a set of tokens and expect exactly one of these tokens to be "E2U". In this way, the ENUM client might be designed to handle both the old and the current forms without added complexity.

To facilitate this method, IANA should reject any request to register an Enumservice with the label "E2U".

To summarise, ENUM clients MUST support ENUM NAPTRs according to [RFC3761] syntax. ENUM clients SHOULD also support ENUM NAPTRs according to the obsolete syntax of [RFC2916]; there are still zones that hold "old" syntax NAPTRs. ENUM zones MUST NOT be provisioned with NAPTRs according to the obsolete form, and MUST be provisioned with NAPTRs in which the Services field is according to [RFC3761].

7. Collected Implications for ENUM Provisioning

ENUM NAPTRs SHOULD NOT include characters outside the printable US-ASCII equivalent range (U+0020 to U+007E) unless it is clear that all ENUM clients they are designed to support will be able to process such characters correctly. If ENUM zone provisioning systems require non-ASCII characters, these systems SHOULD encode the non-ASCII data to emit only US-ASCII characters by applying the appropriate

mechanism ([RFC3492], [RFC3987]). Non-printable characters SHOULD NOT be used, as ENUM clients may need to present NAPTR content in a human-readable form.

The case-sensitivity flag ('i') is inappropriate for ENUM, and SHOULD NOT be provisioned into the Regexp field of E2U NAPTRs.

ENUM zone provisioning systems SHOULD use '!' (U+0021) as their Regexp delimiter character.

If the Regexp delimiter is a character in the static text of the Repl sub-field, it MUST be "escaped" using the escaped-delimiter production of the BNF specification shown in Section 3.2 of [RFC3402] (i.e., "\!", U+005C U+0021). Note that when a NAPTR resource record is entered in DNS master file syntax, the backslash itself must be escaped using a second backslash.

If present in the ERE sub-field of an ENUM NAPTR, the literal character '+' MUST be escaped as "\+" (i.e. U+005C U+002B). Note that, as always, when a NAPTR resource record is entered in DNS master file syntax, the backslash itself must be escaped using a second backslash.

The Registrant and the ENUM zone provisioning system he or she uses SHOULD NOT rely on ENUM clients solely taking account of the value of the ORDER and the PREFERENCE/PRIORITY fields in ENUM NAPTRs. Thus, a Registrant SHOULD place into his or her zone only contacts that he or she is willing to support; even those with the worst ORDER and PREFERENCE/PRIORITY values MAY be selected by an end user.

Many apparent mistakes in ORDER and PREFERENCE/PRIORITY values have been detected in provisioned ENUM zones. To avoid these common interoperability issues, provisioning systems SHOULD NOT use different ORDER field values for NAPTRs in a Resource Record Set (RRSet). To generalise, all ENUM NAPTRs SHOULD hold a default value in their ORDER field. A value of "100" is recommended, as it seems to be used in most provisioned domains.

Multiple NAPTRs with identical ORDER and identical PREFERENCE/ PRIORITY field values SHOULD NOT be provisioned into an RRSet unless the intent is that these NAPTRs are truly identical and there is no preference between them. Implementers SHOULD NOT assume that the DNS will deliver NAPTRs within an RRSet in a particular sequence.

An ENUM zone provisioning system SHOULD assume that, if it generates compound NAPTRs, the Enumservices will normally be processed in left-to-right order within such NAPTRs.

ENUM zone provisioning systems SHOULD assume that, once a non-terminal NAPTR has been selected for processing, the ORDER field value in a domain referred to by that non-terminal NAPTR will be considered only within the context of that referenced domain (i.e., the ORDER value will be used only to sort within the current RRSet and will not be used in the processing of NAPTRs in any other RRSet).

Whilst this client behaviour is non-compliant, ENUM provisioning systems and their users should be aware that some ENUM clients have been detected with poor (or no) support for non-trivial ERE sub-field expressions.

ENUM provisioning systems SHOULD be cautious in the use of multiple back-reference patterns in the Repl sub-field of NAPTRs they provision. Some clients have limited buffer space for character expansion when generating URIs (see also Section 3). These provisioning systems SHOULD check the back-reference replacement patterns they use, ensuring that regular expression processing will not produce excessive-length URIs.

As current support is limited, non-terminal NAPTRs SHOULD NOT be provisioned in ENUM zones unless it is clear that all ENUM clients that this environment supports can process these.

When populating a set of domains with NAPTRs, ENUM zone provisioning systems SHOULD NOT configure non-terminal NAPTRs so that more than 5 such NAPTRs will be processed in an ENUM query.

In a non-terminal NAPTR encountered in an ENUM query (i.e., one with an empty Flags field), the Services field SHOULD be empty.

A non-terminal NAPTR MUST include its target domain in the (non-empty) Replacement field. This field MUST be interpreted as holding the domain name that forms the next key output from this non-terminal rule. The Regexp field MUST be empty in a non-terminal NAPTR intended to be encountered during an ENUM query.

ENUM zones MUST NOT be provisioned with NAPTRs according to the obsolete form, and MUST be provisioned with NAPTRs in which the Services field is according to [RFC3761].

8. Collected Implications for ENUM Clients

ENUM clients SHOULD NOT discard NAPTRs in which they detect characters outside the US-ASCII printable range (0x20 to 0x7E hexadecimal).

ENUM clients MAY discard NAPTRs that have octets in the Flags, Services, or Regexp fields that have byte values outside the US-ASCII equivalent range (i.e., byte values above 0x7F). Clients MUST be ready to encounter NAPTRs with such values without failure.

ENUM clients SHOULD NOT assume that the delimiter is the last character of the Regexp field.

> Unless they are sure that in their environment this is the case, in general an ENUM client may still encounter NAPTRs that have been provisioned with a following 'i' (case-insensitive) flag, even though that flag has no effect at all in an ENUM scenario.

ENUM clients SHOULD discard NAPTRs that have more or less than 3 unescaped instances of the delimiter character within the Regexp field.

> In the spirit of being liberal with what it will accept, if the ENUM client is sure how the Regexp field should be interpreted, then it may choose to process the NAPTR even in the face of an incorrect number of unescaped delimiter characters. If it is not clear how the Regexp field should be interpreted, then the client must discard the NAPTR.

Where the ENUM client presents a list of possible URLs to the end user for his or her choice, it MAY present all NAPTRs -- not just the ones with the highest currently unprocessed ORDER field value. The client SHOULD keep to the ORDER and PREFERENCE/PRIORITY values specified by the Registrant.

ENUM clients SHOULD accept all NAPTRs with identical ORDER and identical PREFERENCE/PRIORITY field values, and process them in the sequence in which they appear in the DNS response. (There is no benefit in further randomising the order in which these are processed, as intervening DNS Servers might have done this already).

ENUM clients receiving compound NAPTRs (i.e., ones with more than one Enumservice) SHOULD process these Enumservices using a left-to-right sort ordering, so that the first Enumservice to be processed will be the leftmost one, and the last will be the rightmost one.

ENUM clients SHOULD consider the ORDER field value only when sorting NAPTRs within a single RRSet. The ORDER field value SHOULD NOT be taken into account when processing NAPTRs across a sequence of DNS queries created by traversal of non-terminal NAPTR references.

ENUM clients MUST be ready to process NAPTRs that use a different character from '!' as their Regexp Delimiter without failure.

ENUM clients MUST be ready to process NAPTRs that have non-trivial patterns in their ERE sub-field values without failure.

ENUM clients MUST be ready to process NAPTRs with a DDDS Application identifier other than 'E2U' without failure.

ENUM clients MUST be ready to process NAPTRs with many copies of back-reference patterns within the Repl sub-field without failure (see also Section 3).

If a NAPTR is discarded, this SHOULD NOT cause the whole ENUM query to terminate and processing SHOULD continue with the next NAPTR in the returned Resource Record Set (RRSet).

When an ENUM client encounters a compound NAPTR (i.e., one containing more than one Enumservice) and cannot process or cannot recognise one of the Enumservices within it, that ENUM client SHOULD ignore this Enumservice and continue with the next Enumservice within this NAPTR's Services field, discarding the NAPTR only if it cannot handle any of the Enumservices contained. These conditions SHOULD NOT be considered errors.

ENUM clients MUST support ENUM NAPTRs according to [RFC3761] syntax. ENUM clients SHOULD also support ENUM NAPTRs according to the obsolete syntax of [RFC2916]; there are still zones that hold "old" syntax NAPTRs.

8.1. Non-Terminal NAPTR Processing

ENUM clients MUST be ready to process NAPTRs with an empty Flags field ("non-terminal" NAPTRs) without failure. More generally, non-terminal NAPTR processing SHOULD be implemented, but ENUM clients MAY discard non-terminal NAPTRs they encounter.

ENUM clients SHOULD ignore any content of the Services field when encountering a non-terminal NAPTR with an empty Flags field.

ENUM clients receiving a non-terminal NAPTR with an empty Flags field MUST treat the Replacement field as holding the domain name to be used in the next round of the ENUM query. An ENUM client MUST discard such a non-terminal NAPTR if the Replacement field is empty or does not contain a valid domain name. By definition, it follows that the Regexp field will be empty in such a non-terminal NAPTR. If present in a non-terminal NAPTR, a non-empty Regexp field MUST be ignored by ENUM clients.

If a problem is detected when processing an ENUM query across multiple domains (by following non-terminal NAPTR references), then the ENUM query SHOULD NOT be abandoned, but instead processing SHOULD continue at the next NAPTR after the non-terminal NAPTR that referred to the domain in which the problem would have occurred.

If all NAPTRs in a domain traversed as a result of a reference in a non-terminal NAPTR have been discarded, then the ENUM client SHOULD continue its processing with the next NAPTR in the "referring" RRSet (i.e., the one including the non-terminal NAPTR that caused the traversal).

ENUM clients MAY consider a chain of more than 5 "non-terminal" NAPTRs traversed in a single ENUM query as an indication that a referential loop has been entered.

Where a domain is about to be entered as the result of a reference in a non-terminal NAPTR, and the ENUM client has detected a potential referential loop, then the client SHOULD discard the non-terminal NAPTR from its processing and continue with the next NAPTR in its list. It SHOULD NOT make the DNS query indicated by that non-terminal NAPTR.

9. Security Considerations

In addition to the security implications of recommendations in this document, those in the basic use of ENUM (and specified in the normative documents for this protocol) should be considered as well; this document does not negate those in any way.

The clarifications throughout this document are intended only as that: clarifications of text in the normative documents. They do not appear to have any security implications above those mentioned in the normative documents.

The suggestions in Section 2, Section 4, and Section 6 do not appear to have any security considerations (either positive or negative).

The suggestions in Section 5.2.2 are a valid approach to a known security threat. It does not open an advantage to an attacker in causing excess processing or memory usage in the client. It does, however, mean that an ENUM client will traverse a "tight loop" of non-terminal NAPTRs in two domains 5 times before the client detects this as a loop; this does introduce slightly higher processing load than would be provided using other methods, but avoids the risks they incur.

As mentioned in Section 3, ENUM uses regular expressions to generate URIs. Though it is a standard feature of DDDS, use of "non-greedy" regular expressions with multiple back-reference patterns in the Repl sub-field does create the potential for buffer-overrun attacks. Provisioning system designers SHOULD be aware of this and SHOULD limit the repeated use of back-reference replacement patterns. Conversely, ENUM client implementers SHOULD avoid using fixed character buffers when generating URIs from Repl sub-fields that include Back-reference patterns, and MUST avoid failure in the case of buffer exhaustion.

10. Acknowledgements

We would like to thank the various development teams who implemented ENUM (both creation systems and clients) and who read the normative documents differently -- without these differences it would have been harder for us all to develop robust clients and suitably conservative management systems. We would also thank those who allowed us to check their implementations to explore behaviour; their trust and help were much appreciated.

In particular, thanks to Richard Stastny for his hard work on a similar task, TS 102 172 [ETSI-TS102172] under the aegis of ETSI, and for supporting some of the ENUM implementations that exist today.

Finally, thanks for the dedication of Michael Mealling in giving us such detailed DDDS specifications, without which the ENUM development effort would have had a less rigorous framework on which to build. This document reflects how complex a system it is: without the intricacy of [RFC3401] - [RFC3404] and the work that went into them, it could have been very difficult to ensure interoperability.

11. References

11.1. Normative References

[E.164] ITU-T, "The International Public Telecommunication Number Plan", Recommendation E.164, February 2005.

[IEEE.1003-2.1992] Institute of Electrical and Electronics Engineers, "Information Technology - Portable Operating System Interface (POSIX) - Part 2: Shell and Utilities (Vol. 1)", IEEE Standard 1003.2, January 1993.

[RFC1034] Mockapetris, P., "Domain names - concepts and facilities", STD 13, RFC 1034, November 1987.

[RFC1035] Mockapetris, P., "Domain names - implementation and specification", STD 13, RFC 1035, November 1987.

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.

[RFC3402] Mealling, M., "Dynamic Delegation Discovery System (DDDS) Part Two: The Algorithm", RFC 3402, October 2002.

[RFC3403] Mealling, M., "Dynamic Delegation Discovery System (DDDS) Part Three: The Domain Name System (DNS) Database", RFC 3403, October 2002.

[RFC3404] Mealling, M., "Dynamic Delegation Discovery System (DDDS) Part Four: The Uniform Resource Identifiers (URI)", RFC 3404, October 2002.

[RFC3405] Mealling, M., "Dynamic Delegation Discovery System (DDDS) Part Five: URI.ARPA Assignment Procedures", BCP 65, RFC 3405, October 2002.

[RFC3490] Faltstrom, P., Hoffman, P., and A. Costello, "Internationalizing Domain Names in Applications (IDNA)", RFC 3490, March 2003.

[RFC3491] Hoffman, P. and M. Blanchet, "Nameprep: A Stringprep Profile for Internationalized Domain Names (IDN)", RFC 3491, March 2003.

[RFC3492] Costello, A., "Punycode: A Bootstring encoding of Unicode for Internationalized Domain Names in Applications (IDNA)", RFC 3492, March 2003.

[RFC3629] Yergeau, F., "UTF-8, a transformation format of ISO 10646", STD 63, RFC 3629, November 2003.

[RFC3761] Faltstrom, P. and M. Mealling, "The E.164 to Uniform Resource Identifiers (URI) Dynamic Delegation Discovery System (DDDS) Application (ENUM)", RFC 3761, April 2004.

[RFC3966] Schulzrinne, H., "The tel URI for Telephone Numbers", RFC 3966, December 2004.

[RFC3986] Berners-Lee, T., Fielding, R., and L. Masinter, "Uniform Resource Identifier (URI): Generic Syntax", STD 66, RFC 3986, January 2005.

[RFC3987] Duerst, M. and M. Suignard, "Internationalized Resource Identifiers (IRIs)", RFC 3987, January 2005.

11.2. Informative References

[ASCII] American National Standards Institute, "Coded Character Set - 7-bit American Standard Code for Information Interchange", ANSI X3.4, 1986.

[ETSI-TS102172] ETSI, "Minimum Requirements for Interoperability of European ENUM Implementations", ETSI TS 102 172, October 2004.

[RFC2915] Mealling, M. and R. Daniel, "The Naming Authority Pointer (NAPTR) DNS Resource Record", RFC 2915, September 2000.

[RFC2916] Faltstrom, P., "E.164 number and DNS", RFC 2916, September 2000.

[RFC3401] Mealling, M., "Dynamic Delegation Discovery System (DDDS) Part One: The Comprehensive DDDS", RFC 3401, October 2002.

[RFC3824] Peterson, J., Liu, H., Yu, J., and B. Campbell, "Using E.164 numbers with the Session Initiation Protocol (SIP)", RFC 3824, June 2004.

Authors' Addresses

Lawrence Conroy
Roke Manor Research
Roke Manor
Old Salisbury Lane
Romsey
United Kingdom

Phone: +44-1794-833666
EMail: lconroy@insensate.co.uk
URI: http://www.sienum.co.uk

Kazunori Fujiwara
Japan Registry Services Co., Ltd.
Chiyoda First Bldg. East 13F
3-8-1 Nishi-Kanda Chiyoda-ku
Tokyo 101-0165
JAPAN

EMail: fujiwara@jprs.co.jp
URI: http://jprs.co.jp/en/