INTERNET-DRAFT
Document: draft-ietf-crisp-firs-dns-03.txt
Expires: March, 2004
Category: Standards-Track

Eric A. Hall

August 2003

# Defining and Locating DNS Domains in the Federated Internet Registry Service

## Status of this Memo

This document is an Internet-Draft and is in full conformance with all provisions of Section 10 of RFC 2026.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF), its areas, and its working groups. Note that other groups may also distribute working documents as Internet-Drafts.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

The list of current Internet-Drafts can be accessed at http://www.ietf.org/ietf/1id-abstracts.txt

The list of Internet-Draft Shadow Directories can be accessed at http://www.ietf.org/shadow.html.

## Copyright Notice



## Abstract

This document defines LDAP schema and searching rules for DNS domain names, in support of the Federated Internet Registry Service (FIRS) described in [FIRS-ARCH] and [FIRS-CORE].



## Table of Contents

## 1. Introduction

This specification defines the naming syntax, object classes, attributes, matching filters, and query processing rules for storing and locating DNS domain names in the FIRS service. Refer to [FIRS-ARCH] for information on the FIRS architecture and [FIRS-CORE] for the schema definitions and rules which govern the FIRS service as a whole.

Note that these rules and definitions only apply to domain name resources, and do not apply to domainComponent entries or any other domain name elements, unless explicitly defined. Also note that this specification governs reverse-lookup DNS domains for IPv4 and IPv6 address blocks, but that these entries are entirely different from the entries which govern the actual IPv4 and IPv6 address blocks themselves.

The definitions in this specification are intended to be used with FIRS. Their usage outside of FIRS is not prohibited, but any such usage is beyond this specification's scope of authority.

## 2. Prerequisites and Terminology

The complete set of specifications in the FIRS collection cumulative define a structured and distributed information service using LDAPv3 for the data-formatting and transport functions. This specification should be read in the context of that set, which currently includes [FIRS-ARCH], [FIRS-CORE], [FIRS-DNSRR], [FIRS-CONTCT], [FIRS-ASN], [FIRS-IPV4] and [FIRS-IPV6].

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 RFC 2119.

## 3. Naming Syntax

The naming syntax for DNS domains in FIRS MUST follow the form of "cn=<inetDnsDomainSyntax>,cn=inetResources,<partition>", where <inetDnsDomainSyntax> is the DNS domain name resource, and where <partition> is a sequence of domainComponent relative distinguished names which identifies the scope of authority for the selected directory partition.

The inetDnsDomainSyntax syntax is as follows:

```
inetDnsDomainSyntax
( 1.3.6.1.4.1.7161.1.3.0
  NAME 'inetDnsDomainSyntax'
  DESC 'A fully-qualified DNS domain name.' )
```

The inetDnsDomainSyntax uses relatively unstructured UTF-8 strings, using standardized procedures to produce heavily-normalized DNS domain names rather than using formal domain name syntax rules. This is partly necessary due to conflicting syntax rules in the different base specifications, but is also necessary in order to support existing LDAP systems which do not know the syntax rules.

Section 3.1 defines the normalization and conversion process which is used to produce the standardized output. All systems which generate DNS domain names for use with FIRS MUST use these normalization and conversion process on those domain names.

### 3.1. Normalization and Conversion

The normalization and conversion routine described herein produce UTF-8 [RFC2279] encoded domain names as output, with the resulting sequences being suitable for equality matches, sub-string matches, and a broad range of other matching operations. Once all of these steps have successfully completed, the domain name can be stored in the directory or used as an assertion value. Any fatal error conditions encountered during these conversions MUST result in a local failure; FIRS-aware applications MUST NOT store or transmit non-normalized domain names for any purposes.

> NOTE: The use of UTF-8 encoded domain names is ONLY required for protocol-level exchanges of domain name resources. Clients MAY use any encoding or transformation formats that they wish for local presentation services. Specifically, these requirements are intended to ensure interoperability between clients and servers, and do not mandate any presentation format at the client.

In general terms, the validation process requires that every domain name which is to be stored in an internationalized domain name element undergo a two-part conversion, with the input first being reduced to its canonical IDNA-encoded form, and then being expanded into its UTF-8 encoded UCS form. This process ensures that the domain name has been validated as a semantically correct IDNA sequence, and that the resulting internationalized domain name has been properly normalized into its canonical form.

The full process is as follows:

a. Unless otherwise explicitly defined, disable the UseSTD3ASCIIRules IDNA flag and enable the AllowUnassigned IDNA flag, thereby permitting the broadest range of character codes to be used.

b. If the input domain name terminates with a Full-Stop character (0x2E), an Ideographic Full-Stop (U+3002), Full-Width Full-Stop (U+FF0E) character, or a Half-Width Ideographic Full-Stop (U+FF61), but does not consist of that single character alone, remove the trailing character from the input.

c. If the input domain name contains any octet values which need to be protected from normalization, use the escape syntax described in section 3.2 to protect those octets.

d. Perform the "ToASCII" conversion operation specified in [RFC3490]. This step will reduce the input domain name to the canonical IDNA-compatible form, thus ensuring that the input data can be properly normalized when it is reconstructed, and also ensuring that any subsequent conversions back into the ASCII-compatible form will result in predictable and legitimate domain names.

e. Perform the "ToUnicode" conversion operation specified in [RFC3490] against the output from step 3.1.d above. This step will convert the ASCII-compatible sequence into a sequence of UCS code-point values.

f. Encode the output from step 3.1.e into UTF-8.

Note that the UseSTD3ASCIIRules and AllowUnassigned IDNA flags MUST be set to their most liberal settings by default, and are not to be used unless the underlying application-specific usage of a domain name is known to require usage to the contrary.

By following these rules, internationalized domain names will always be valid, and will always be usable by applications which specifically make use of the elements, while those systems which do not make explicit use of these elements but which may inadvertently pass the internationalized domain names to other applications will not be exposed to any potential risks which could have been caused by malformed data.

Also note that these requirements are significantly more stringent than the requirements for validating legacy domain names in the legacy elements, and also apply to legacy-compatible domain names which are stored in the internationalized elements. For example, the existing domainComponent and mail attributes do not require data to be validated against the known syntax rules for domain names and email addresses, but instead simply limit the range of character codes to a relatively small subset, while the rules defined above will result in the same canonical input having a stricter actual syntax.

Also note that UTF-8 character codes are frequently illegal as data in URLs, and many of those octet values will probably be escaped before they are stored in a URL as data.

### 3.2. Escape Syntax

Certain applications allow for the use of "unusual" characters or octet values which are not typically associated with traditional domain names, but which must be preserved in order for the associated applications to function properly. For example, an application-specific domain name may contain an Underscore character (0x5F) or a Space character (0x20), or may contain a "raw" octet value such as 0xC0 which cannot be treated as a UCS character code during normalization routines (otherwise the corresponding UCS character code value would be interpreted and lowercased, thus destroying the actual octet value).

In order to ensure that these kinds of values are properly preserved, a formal escape syntax is defined for their use. In general terms, this syntax requires problematic eight-bit values to be replaced with a Reverse-Solidus character (0x5C, "\"), followed by a three-digit decimal value (in the range of "000" through "255") that corresponds to the canonical octet value.

This escape syntax MUST be applied to any octet value which does not explicitly represent a printable character (0x00 through 0x20, 0x7F through 0x9F, and 0xA0, inclusive), or which represents an embedded Reverse-Solidus character (0x5C, "\"). In those cases where a valid escape sequence already exists, that sequence (including its leading Reverse-Solidus character) MUST NOT be escaped again.

This escape syntax MAY be applied to any other character code or octet value, although the unnecessary usage of this mechanism is strongly DISCOURAGED. Furthermore, the availability of this mechanism MUST NOT be interpreted to mean that this mechanism can be used with any domain name; instead, it is only to be used with application-specific domain names which explicitly allow the presence of these problematic characters.

For example, if an application-specific domain name contains "weird name.example.com", the "weird name" portion of that domain name MUST be escaped as "weird\032name". Meanwhile, if an application-specific domain name contains "local\046postmaster", this sequence would be unmodified since the Reverse-Solidus character is already part of a valid escape sequence.

This escape syntax MUST be applied to an input domain name before that domain name undergoes the conversion process described in

section 3.1. Furthermore, the leaf-node applications which generate and use these domain names SHOULD escape the data before it is passed to an LDAP agent, since those agents cannot be expected to know all of the application-specific usages of a domain name. For example, an application which uses a domain name with an embedded Full-Stop character (0x2E, ".") SHOULD escape that character before storing or passing the domain name to an LDAP agent, thus eliminating the possibility of having that agent interpret the embedded Full-Stop character as a label separator.

Note that any Reverse Solidus characters in the resulting domain name will be further escaped when these sequences are transferred in LDAP messages. For example, "weird\032name" will be further escaped as "weird\\032name" when it is passed in an LDAP message (this secondary escape will be stripped upon receipt, leaving the escaped domain name in its original form).

Also note that Reverse-Solidus characters are frequently illegal as data in URIs, and these characters will probably end up being percent-escaped whenever they are provided in a URI as data.

## 4. Object Classes and Attributes

DNS domain name entries in FIRS MUST use the inetDnsDomain object class, in addition to the mandatory object classes defined in [FIRS-CORE]. DNS domain name entries MUST be treated as containers capable of holding subordinate entries.

If an entry exists as a referral source, the entry MUST be defined with the referral object class, in addition to the other object classes defined above. Referral sources MUST NOT contain subordinate entries. Refer to section 3.5 of [FIRS-CORE] for more information on referral entries in FIRS.

The inetDnsDomain object class is a structural object class which is subordinate to the inetResources object class. The inetDnsDomain object class has no mandatory attributes, although it does have several optional attributes. The inetDnsDomain object class also inherits the attributes defined in the inetResources object class, including the "cn" naming attribute.

Domain name entries MAY also be defined with the inetDnsRR auxiliary object class (as described in [FIRS-DNSRR]), which provides DNS resource records as attributes. For example, if a domain name entry needs to publish a list of authoritative DNS servers for the associated domain name, those values would be

provided through the use of the inetDnsRR object class and its related attributes.

The schema definition for the inetDnsDomain object class is as follows:

```
inetDnsDomain
( 1.3.6.1.4.1.7161.1.3.1
  NAME 'inetDnsDomain'
  DESC 'DNS domain attributes.'
  SUP inetResources
  STRUCTURAL
  MAY ( inetDnsDelegationStatus $ inetDnsDelegationDate $
   inetDnsRegistrar $ inetDnsRegistry $ inetDnsContacts ) )
```

The attributes from the inetDnsDomain object class are described below:

```
inetDnsContacts
( 1.3.6.1.4.1.7161.1.3.2
  NAME 'inetDnsContacts'
  DESC 'Contacts for general administrative issues concerning
  this domain name.'
  EQUALITY caseIgnoreMatch
  SYNTAX 1.3.6.1.4.1.7161.1.4.0 )

inetDnsDelegationDate
( 1.3.6.1.4.1.7161.1.3.3
  NAME 'inetDnsDelegationDate'
  DESC 'Date this DNS domain name was delegated.'
  EQUALITY generalizedTimeMatch
  ORDERING generalizedTimeOrderingMatch
  SYNTAX 1.3.6.1.4.1.1466.115.121.1.24
  SINGLE-VALUE )

inetDnsDelegationStatus
( 1.3.6.1.4.1.7161.1.3.4
  NAME 'inetDnsDelegationStatus'
  DESC 'Delegation status of this domain name.'
  EQUALITY numericStringMatch
  SYNTAX 1.3.6.1.4.1.1466.115.121.1.36{2}
  SINGLE-VALUE )
```

NOTE: In an effort to facilitate internationalization and programmatic processing, the current status of a delegation is identified by a 16-bit integer. The values and status mapping is as follows:

```
0   Reserved delegation (permanently inactive)
1   Assigned and active (normal state)
2   Assigned but not yet active (new delegation)
3   Assigned but on hold (disputed)
4   Assignment revoked (database purge pending)
```

Additional values are reserved for future use, and are to be administered by IANA.

Note that there is no status code for "unassigned"; unassigned entries SHOULD NOT exist, and SHOULD NOT be returned as answers.

```
inetDnsRegistrar
( 1.3.6.1.4.1.7161.1.3.5
  NAME 'inetDnsRegistrar'
  DESC 'Registrar who delegated this domain name.'
  EQUALITY caseExactMatch
  SYNTAX 1.3.6.1.4.1.1466.115.121.1.15 )
```

NOTE: The inetDnsRegistrar attribute uses a URL to indicate the registrar who delegated the domain name. The attribute structure is identical to the labeledURI attribute, as defined in [RFC2798], including the URL and textual comments. The data can refer to any valid URL.

```
inetDnsRegistry
( 1.3.6.1.4.1.7161.1.3.6
  NAME 'inetDnsRegistry'
  DESC 'Registry where this domain name is managed.'
  EQUALITY caseExactMatch
  SYNTAX 1.3.6.1.4.1.1466.115.121.1.15 )
```

NOTE: The inetDnsRegistry attribute uses a URL to indicate the registry who is ultimately responsible for the domain name. The attribute structure is identical to the labeledURI attribute, as defined in [RFC2798], including the URL and textual comments. The data can refer to any valid URL.

Two examples of the inetDnsDomain object class are shown below. The examples also include attributes from the inetResources and referral object classes.

```
cn=example.com,cn=inetResources,dc=com
[top object class]
[inetResources object class]
[inetDnsDomain object class]
|
+-attribute: description
| value: "The example.com DNS domain"
|
+-attribute: inetDnsContacts
| value: "hostmaster@example.com"
|
+-cn=ref1,cn=example.com,cn=inetResources,dc=com
  [top object class]
  [inetResources object class]
  [inetDnsDomain object class]
  [referral object class]
  |
  +-attribute: ref
    value: "ldap:///dc=registrar,dc=com???
           (1.3.6.1.4.1.7161.1.3.0.1:=example.com)"
```

Figure 1: The entry for the example.com DNS domain name in the dc=com partition, and a referral child entry.

## 5. Query Processing Rules

Queries for DNS domain names have several special requirements, as discussed in the following sections.

Refer to [FIRS-CORE] for general information about FIRS queries.

### 5.1. Query Pre-Processing

FIRS clients MUST use the top-down bootstrap model by default for DNS domain name queries. As such, the search base for default queries would be set to the right-most domainComponent relative distinguished name of the authoritative partition, rather than being set to the fully-qualified distinguished name of the authoritative partition.

FIRS clients MAY use the targeted or bottom-up bootstrap models for queries if necessary or desirable. However, it is not likely

that entries will be found for all DNS domain name resources using these models. As such, the top-down bootstrap model will be the most useful in most cases, and MUST be used by default.

When the bottom-up bootstrap model is used, the authoritative partition for a DNS domain name is determined by mapping the normalized domain name to a sequence of domainComponent labels.

As a simple example, "www.example.com" would be mapped to the "dc=www,dc=example,dc=com" authoritative partition, with this partition being used to seed the query process. As a slightly more complex example, the domain name of "weird name.example.com" would be mapped to "dc=weird\032name,dc=example,dc=com".

Since the domainComponent attribute is restricted to seven-bit characters, the normalized DNS domain name MUST be converted to its IDNA form using the "ToASCII" conversion operation specified in [RFC3490] before these lookups are performed, with the "UseSTD3ASCIIRules" flag disabled (FIRS applications MAY reuse the output from the conversion performed in step 3.1.d if the entire conversion process is known to have completed successfully). The resulting sequence of ASCII labels are used to form the domainComponent sequence which represents the authoritative partition for the DNS domain name.

## 5.2. LDAP Matching

If the server advertises the inetDnsDomain object class and the inetDnsDomainMatch matching filter in the inetResourcesControl server control, FIRS clients MUST use the inetDnsDomainMatch matching filter in LDAP searches for DNS domain name entries.

The inetDnsDomainMatch filter provides an identifier and search string format which collectively inform a queried server that a specific DNS domain name should be searched for, and that any inetDnsDomain object class entries which either match or are delegation parents to the assertion value should be returned.

The inetDnsDomainMatch filter is defined as follows:

```
inetDnsDomainMatch
( 1.3.6.1.4.1.7161.1.3.0.1
  NAME 'inetDnsDomainMatch'
  SYNTAX 1.3.6.1.4.1.7161.1.3.0 )
```

Clients MUST ensure that the query input is normalized according to the rules specified in section 3 before the input is used as the assertion value to the resulting LDAP query.

A FIRS server MUST compare the assertion value against the distinguished name of all entries within and beneath the container specified by the search base of the query. Any entry in that hierarchy with an object class of inetDnsDomain and a distinguished name component that is either equal to or is a delegation parent of the domain name provided in the assertion value MUST be returned to the client (this specifically includes any child entries, such as referral stubs). Entries which do not have an object class of inetDnsDomain MUST NOT be returned. Entries with distinguished name for other delegation hierarchies MUST NOT be returned. Entries with distinguished names for child domains MUST NOT be returned.

An example of this matching logic is illustrated below, using the assertion value of "example.com" and the search base of "cn=inetResources,dc=com":

```
set searchBase "cn=inetResources,dc=com"
find ( ( objectClass equals inetDnsDomain) and
        ( ( nameComponent equals "cn=com" ) or
          ( nameComponent equals "cn=example.com") )
```

Domain names MUST be compared on label boundaries, and MUST NOT be compared through simple character matching. Given two entries of "cn=example.com" and "cn=an-example.com", only the first would match an assertion value of "example.com".

Note that the entry name of "cn=." encompasses the entire DNS domain namespace. When used in conjunction with referrals, this entry MAY be used to redirect all inetDnsDomainMatch queries to another partition for subsequent processing.

The matching filters defined in this specification MUST be supported by FIRS clients and servers. FIRS servers MAY support additional matching filters, although FIRS clients MUST NOT expect any additional filters to be available.

If the server does not advertise support for the inetDnsDomainMatch matching filter in the inetResourcesControl server control, the client MAY choose to emulate the matching filter through the use of locally-constructed equalityMatch filters. However, this process can result in incomplete answers in

some cases, so if the server advertises support for the inetDnsDomainMatch matching filter in the inetResourcesControl control, the client MUST use it.

### 5.3. Example Query

The following example assumes that the user has specified "www.example.com" as the query value:

a. Normalize the input, which is "www.example.com" in this case.

b. Determine the authoritative partition, which is "dc=www,dc=example,dc=com" in this case. By default, queries for DNS domain names use the top-down model, meaning that the right-most relative distinguished name of "dc=com" will be used.

c. Determine the search base for the query, which will be "cn=inetResources,dc=com" if the defaults are used.

d. Initiate a DNS lookup for the SRV resource records associated with "_ldap._tcp.com." For the purpose of this example, assume that this lookup succeeds, with the DNS response message indicating that "firs.iana.org" is the preferred LDAP server.

e. Submit an LDAPv3 query to the specified server, using "(1.3.6.1.4.1.7161.1.3.0.1:=www.example.com)" as the matching filter, "cn=inetResources,dc=com" as the search base, and the global query defaults defined in [FIRS-CORE].

f. Assume that the queried server returns a continuation reference referral which points to "ldap:///cn=inetResources,dc=netsol,dc=com". The distinguished name element of "cn=inetResources,dc=netsol,dc=com" will be used as the new search base, while "dc=netsol,dc=com" will be used as the new authoritative partition.

g. Initiate a DNS lookup for the SRV resource records associated with "_ldap._tcp.netsol.com." For the purpose of this example, assume that this lookup succeeds, with the DNS response message indicating that "firs.netsol.org" is the preferred LDAP server.

h. Submit an LDAPv3 query to the specified server, using "(1.3.6.1.4.1.7161.1.3.0.1:=www.example.com)" as the matching filter, "cn=inetResources,dc=netsol,dc=com" as the search base, and the global query defaults defined in [FIRS-CORE].

i. Assume that no other referrals are received. Display the answer data which has been received and exit the query.

## 6. Variant Domain Names

Some domain operators have policies which require that variant forms of a domain name be assigned or reserved whenever the underlying domain name is registered. For example, a domain operator may choose to reserve look-alike forms of "foo" (including "f00" and "fo0" and so forth), thereby preventing other entities from registering the look-alike domain name.

This document reserves the inetDnsDelegationStatus attribute value of "5" specifically for use with the look-alike domains. In this model, the canonical domain name would have a typical entry, while all of the look-alike domains would have entries with the inetDnsDelegationStatus attribute value of "5", and would only exist as referrals to the canonical domain name's entry. Searches and lookups for the variant domain names would return referrals which point to the canonical domain name entry.

An entry for the canonical domain name MUST exist in the appropriate partition(s). These entries MAY include the variant domain names as values of the optional inetAssociatedDnsDomains attribute, if desired.

## 7. Security Considerations

Security considerations are discussed in [FIRS-ARCH].

## 8. IANA Considerations

This specification assumes the existence of partitions for each of the top-level domain names in the global DNS namespace, with the expectation that FIRS-capable LDAP servers will be established for each of these partitions, and with these partition containing domain delegation entries which will provide referrals to the appropriate registrar's partitions. It is expected that IANA will encourage top-level domain registry operators to oversee the creation and management of these resources.

It is further expected that IANA will oversee the creation and management of the root domain's LDAP SRV resource records, the "dc=." LDAP partition, and the necessary LDAP servers.

The inetDnsDelegationStatus attribute uses numeric code values. It is expected that IANA will manage the assignment of these values.

Additional IANA considerations are discussed in [FIRS-ARCH].

## 9. Normative References

[FIRS-ARCH] Hall, E. "The Federated Internet Registry Service: Architecture and Implementation Guide", draft-ietf-crisp-firs-arch-03, August 2003.

[FIRS-ASN] Hall, E. "Defining and Locating Autonomous System Numbers in the Federated Internet Registry Service", draft-ietf-crisp-firs-asn-03, August 2003.

[FIRS-CONTCT] Hall, E. "Defining and Locating Contact Persons in the Federated Internet Registry Service", draft-ietf-crisp-firs-contact-03, August 2003.

[FIRS-CORE] Hall, E. "The Federated Internet Registry Service: Core Elements", draft-ietf-crisp-firs-core-03, August 2003.

[FIRS-DNSRR] Hall, E. "Defining and Locating DNS Resource Records in the Federated Internet Registry Service", draft-ietf-crisp-firs-dnsrr-02, July 2003.

[FIRS-IPV4] Hall, E. "Defining and Locating IPv4 Address Blocks in the Federated Internet Registry Service", draft-ietf-crisp-firs-ipv4-03, August 2003.

[FIRS-IPV6] Hall, E. "Defining and Locating IPv6 Address Blocks in the Federated Internet Registry Service", draft-ietf-crisp-firs-ipv6-03, August 2003.

[RFC2181] Elz, R., and Bush, R. "Clarifications to the DNS Specification", RFC 2181, July 1997.

[RFC2247] Kille, S., Wahl, M., Grimstad, A., Huber, R., and Sataluri, S. "Using Domains in LDAP/X.500 DNs", RFC 2247, January 1998.

[RFC2251] Wahl, M., Howes, T., and Kille, S. "Lightweight Directory Access Protocol (v3)", RFC 2251, December 1997.

[RFC2252] Wahl, M., Coulbeck, A., Howes, T., and Kille, S. "Lightweight Directory Access Protocol (v3): Attribute Syntax Definitions", RFC 2252, December 1997.

[RFC2254] Howes, T. "The String Representation of LDAP Search Filters", RFC 2254, December 1997.

[RFC2279] Yergeau, F. "UTF-8, a transformation format of ISO 10646", RFC 2279, January 1998.

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

[STD13] Mockapetris, P. "Domain names - concepts and facilities", STD 13, RFC 1034 and "Domain names - implementation and specification", STD 13, RFC 1035, November 1987.

[US-ASCII] Cerf, V. "ASCII format for Network Interchange", RFC 20, October 1969.

## 10. Changes from Previous Versions

draft-ietf-crisp-firs-dns-03:

* Several clarifications and corrections have been made.

* The normalization rules were rewritten to be more exacting and precise.

* Clarified the matching behavior, and added sample logic that demonstrates efficient matching behavior.

* The inetDnsAuthServers attribute was removed. Name servers for a domain resource should be listed using the inetDnsRR object class instead.

* Several attributes had their OIDs changed. NOTE THAT THIS IS AN INTERNET DRAFT, AND THAT THE OIDS ARE SUBJECT TO ADDITIONAL CHANGES AS THIS DOCUMENT IS EDITED.

draft-ietf-crisp-firs-dns-02:

* Several clarifications and corrections have been made.

* Several attributes had their OIDs changed. NOTE THAT THIS IS AN INTERNET DRAFT, AND THAT THE OIDS ARE SUBJECT TO ADDITIONAL CHANGES AS THIS DOCUMENT IS EDITED.

draft-ietf-crisp-firs-dns-01:

* Several clarifications and corrections have been made.

draft-ietf-crisp-firs-dns-00:

* Restructured the document set.

* "Attribute references" have been eliminated from the specification. All referential attributes now provide actual data instead of URL pointers to data. Clients that wish to retrieve these values will need to start new queries using the data values instead of URLs.

* The various modified* operational attributes have been eliminated as unnecessary.

* Several attributes had their OIDs changed. NOTE THAT THIS IS AN INTERNET DRAFT, AND THAT THE OIDS ARE SUBJECT TO ADDITIONAL CHANGES AS THIS DOCUMENT IS EDITED.

draft-ietf-crisp-lw-dns-01:

* Added discussion for internationalized domain names.

* Moved attribute-specific security requirements to the Security section.

## 11. Author's Address

Eric A. Hall
ehall@ehsco.com

## 12. Acknowledgments

Funding for the RFC editor function is currently provided by the Internet Society.

Portions of this document were funded by Verisign Labs.

The first version of this specification was co-authored by Andrew Newton of Verisign Labs, and subsequent versions continue to be developed with his active participation.

## 13. Full Copyright Statement