Network Working Group
Request for Comments: 3375
Category: Informational

S. Hollenbeck
Verisign, Inc.

September 2002

Generic Registry-Registrar Protocol Requirements

Status of this Memo

Copyright Notice

Abstract

This document describes high-level functional and interface requirements for a client-server protocol for the registration and management of Internet domain names in shared registries. Specific technical requirements detailed for protocol design are not presented here. Instead, this document focuses on the basic functions and interfaces required of a protocol to support multiple registry and registrar operational models.

Conventions Used In This Document

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].

Table of Contents

1. Introduction

The advent of shared domain name registration systems illustrates the utility of a common, generic protocol for registry-registrar interaction. A standard generic protocol will allow registrars to communicate with multiple registries through a common interface, reducing operational complexity. This document describes high level functional and interface requirements for a generic provisioning protocol suitable for registry-registrar operations. Detailed technical requirements are not addressed in this document.

1.1 Definitions, Acronyms, and Abbreviations

ccTLD: Country Code Top Level Domain. ".us" is an example of a ccTLD.

DNS: Domain Name System

gTLD: Generic Top Level Domain. ".com" is an example of a gTLD.

IANA: Internet Assigned Numbers Authority

IETF: Internet Engineering Task Force

IP Address: Either or both IPv4 or IPv6 address.

IPv4: Internet Protocol version 4

IPv6: Internet Protocol version 6

RRP: Registry-Registrar Protocol

TLD: Top Level Domain. A generic term used to describe both gTLDs and ccTLDs that exist under the top-level root of the domain name hierarchy.

Exclusive Registration System: A domain name registration system in which registry services are limited to a single registrar. Exclusive Registration Systems are either loosely coupled (in which case the separation between registry and registrar systems is readily evident), or tightly coupled (in which case the separation between registry and registrar systems is obscure).

Name Space: The range of values that can be assigned within a particular node of the domain name hierarchy.

Object: A generic term used to describe entities that are created, updated, deleted, and otherwise managed by a generic registry-registrar protocol.

Registrant: An entity that registers domain names in a registry through the services provided by a registrar. Registrants include individuals, organizations, and corporations.

Registrar: An entity that provides front-end domain name registration services to registrants, providing a public interface to registry services.

Registry: An entity that provides back-end domain name registration services to registrars, managing a central repository of information associated with domain name delegations. A registry is typically responsible for publication and distribution of zone files used by the Domain Name System.

Shared Registration System: A domain name registration system in which registry services are shared among multiple independent registrars. Shared Registration Systems require a loose coupling between registrars and a registry.

Thick Registry: A registry in which all of the information associated with registered entities, including both technical information (information needed to produce zone files) and social information (information needed to implement operational, business, or legal practices), is stored within the registry repository.

Thin Registry: A registry in which all elements of the social information associated with registered entities is distributed between a shared registry and the registrars served by the registry.

Zone: The complete set of information for a particular "pruned" subtree of the domain space. The zone concept is described fully in [RFC1035].

2. General Description

A basic understanding of domain name registration systems provides focus for the enumeration of functional and interface requirements of a protocol to serve those systems. This section provides a high-level description of domain name registration systems to provide context for the requirements identified later in this document.

2.1 System Perspective

A domain name registration system consists of a protocol and associated software and hardware that permits registrars to provide Internet domain name registration services within the name spaces administered by a registry. A registration system can be shared among multiple competing registrars, or it can be served by a single registrar that is either tightly or loosely coupled with back-end registry services. The system providing registration services for the .com, .net, and .org gTLDs is an example of a shared registration system serving multiple competing registrars. The systems providing registration services for some ccTLDs and the .gov and .mil gTLDs are examples of registration systems served by a single registrar.

2.2 System Functions

Registrars access a registry through a protocol to register objects and perform object management functions. Required functions include session management; object creation, update, renewal, and deletion; object query; and object transfer.

A registry generates DNS zone files for the name spaces it serves. Zone files are created and distributed to a series of name servers that provide the foundation for the domain name system.

2.3 User Characteristics

Protocol users fall into two broad categories: entities that use protocol client implementations and entities that use protocol server implementations, though an entity can provide both client and server services if it provides intermediate services. A protocol provides a loose coupling between these communicating entities.

2.4 Assumptions

There is one and only one registry that is authoritative for a given name space and zone.

A registry can be authoritative for more than one name space and zone. Some registry operations can be billable. The impact of a billable operation can be mitigated through the specification of non-billable operations that allow a registrar to make informed decisions before executing billable operations.

A registry can choose to implement a subset of the features provided by a generic registry-registrar protocol. A thin registry, for example, might not provide services to register social information. Specification of minimal implementation compliance requirements is thus an exercise left for a formal protocol definition document that addresses the functional requirements specified here.

A protocol that meets the requirements described here can be called something other than "Generic Registry Registrar Protocol".

The requirements described in this document are not intended to limit the set of objects that can be managed by a generic registry-registrar protocol.

3. Functional Requirements

This section describes functional requirements for a registry-registrar protocol. Technical requirements that describe how these requirements are to be met are out of scope for this document.

3.1 Session Management

[1] The protocol MUST provide services to explicitly establish a client session with a registry server.

[2] In a connection-oriented environment, a server MUST respond to connection attempts with information that identifies the server and the default server protocol version.

[3] The protocol MUST provide services that allow a client to request use of a specific protocol version as part of negotiating a session.

[4] The protocol MUST provide services that allow a server to decline use of a specific protocol version as part of negotiating a session.

[5] A session MUST NOT be established if the client and server are unable to reach agreement on the protocol version to be used for the requested session.

[6] The protocol MUST provide services to explicitly end an established session.

[7] The protocol MUST provide services that provide transactional atomicity, consistency, isolation, and durability in the advent of session management failures.

[8] The protocol MUST provide services to confirm that a transaction has been completed if a session is aborted prematurely.

3.2 Identification and Authentication

[1] The protocol or another layered protocol MUST provide services to identify registrar clients and registry servers before granting access to other protocol services.

[2] The protocol or another layered protocol MUST provide services to authenticate registrar clients and registry servers before granting access to other protocol services.

[3] The protocol or another layered protocol MUST provide services to negotiate an authentication mechanism acceptable to both client and server.

3.3 Transaction Identification

[1] Registry operations that create, modify, or delete objects MUST be associated with a registry-unique identifier. The protocol MUST allow each transaction to be identified in a permanent and globally unique manner to facilitate temporal ordering and state management services.

3.4 Object Management

This section describes requirements for object management, including identification, registration, association, update, transfer, renewal, deletion, and query.

3.4.1 Object Identification

Some objects, such as name servers and contacts, have utility in multiple registries. However, maintaining disjoint copies of object information in multiple registries can lead to inconsistencies that have adverse consequences for the Internet. For example, changing a name server name in one registry, but not in a second registry that refers to the server for domain name delegation, can produce unexpected DNS query results.

[1] The protocol MUST provide services to associate an object identifier with every object.

[2] Object identifiers MUST be globally unique.

[3] An object's identifier MUST NOT change during the lifetime of the object in a particular repository, even if administrative control of the object changes over time.

[4] An object identifier MUST contain information that unambiguously identifies the object.

[5] Object identifier format specified by the protocol SHOULD be easily parsed and understood by humans.

[6] An object's identifier MUST be generated and stored when an object is created.

3.4.2 Object Registration

[1] The protocol MUST provide services to register Internet domain names.

[2] The protocol MUST permit a starting and ending time for a domain name registration to be negotiated, thereby allowing a registry to implement policies allowing a range of registration validity periods (the start and end points in time during which one normally assumes that an object will be active), and enabling registrars to select a period for each registration they submit from within the valid range based on out-of-band negotiation between the registrar and the registrant. Registries SHOULD be allowed to accept indefinitely valid registrations if the policy that they are implementing permits, and to specify a default validity period if one is not selected by a registrar. Registries MUST be allowed to specify minimal validity periods consistent with prevailing or preferred practices for fee-for-service recovery. The protocol MUST provide features to ensure that both registry and registrar have a mutual understanding of the validity period at the conclusion of a successful registration event.

[3] The protocol MUST provide services to register name servers. Name server registration MUST NOT be limited to a specific period of time. Name servers MUST be registered with a valid IPv4 or IPv6 address when a "glue record" is required for delegation. A name server MAY be registered with multiple IP addresses. Multiple name servers using distinct server names MAY share an IP address.

[4] The protocol MUST provide services to manage delegation of zone authority. Names of name servers MUST NOT be required to be tied to the name of the zone(s) for which the server is authoritative.

[5] The protocol MUST provide services to register social information describing human and organizational entities. Registration of social information MUST NOT be limited to a specific period of time. Social information MAY include a name (individual name, organization name, or both), address (including street address, city, state or province (if applicable), postal code, and country), voice telephone number, email address, and facsimile telephone number.

[6] Protocol services to register an object MUST be available to all authorized registrars.

3.4.3 Object Association

[1] The protocol MUST provide services to associate name servers with domain names to delegate authority for zones. A domain name MAY have multiple authoritative name servers. Name servers MAY be authoritative for multiple zones.

[2] The protocol MUST provide services to associate IP addresses with name servers. A name server MAY have multiple IP addresses. An IP address MAY be associated with multiple name server registrations.

[3] The protocol MUST provide services to associate social information with other objects. Social information associations MUST be identified by type. "Registrant" is an example social information type that might be associated with an object such as a domain name.

[4] The protocol MUST provide services to associate object management capabilities on a per-registrar basis.

[5] Some managed objects represent shared resources that might be referenced by multiple registrars. The protocol MUST provide services that allow a registrar to associate an existing shared resource object with other registered objects sponsored by a second registrar. For example, authority for the example.tld zone (example.tld domain object managed by registrar X) and authority for the test.tld zone (test.tld domain object managed by registrar Y) might be delegated to server ns1.example.tld (managed by registrar X). Registrar X maintains administrative control over domain object example.tld and server object ns1.example.tld, and registrar Y maintains administrative control over domain object test.tld. Registrar Y does not have administrative control over server object ns1.example.tld.

3.4.4 Object Update

[1] The protocol MUST provide services to update information associated with registered Internet domain names.

[2] The protocol MUST provide services to update information associated with registered name servers.

[3] The protocol MUST provide services to update social information associated with registered human and organizational entities.

[4] The protocol MUST provide services to limit requests to update a registered object to the registrar that currently sponsors the registered object.

[5] The protocol MUST provide services to explicitly reject unauthorized attempts to update a registered object.

3.4.5 Object Transfer

[1] The protocol MUST provide services to transfer domain names among authorized registrars. Name servers registered in a domain being transferred MUST be transferred along with the domain itself. For example, name servers "ns1.example.tld" and "ns2.example.tld" MUST be implicitly transferred when domain "example.tld" is transferred.

[2] The protocol MUST provide services to describe all objects, including associated objects, that are transferred as a result of an object transfer.

[3] The protocol MUST provide services to transfer social information objects among authorized registrars.

[4] Protocol transfer requests MUST be initiated by the registrar who wishes to become the new administrator of an object.

[5] The protocol MUST provide services to confirm registrar authorization to transfer an object.

[6] The protocol MUST provide services that allow the requesting registrar to cancel a requested object transfer before the request has been approved or rejected by the original sponsoring registrar. Requests to cancel the transfer of registered objects MUST be limited to the registrar that requested transfer of the registered object. Unauthorized attempts to cancel the transfer of a registered object MUST be explicitly rejected.

[7] The protocol MUST provide services that allow the original sponsoring registrar to approve or reject a requested object transfer. Requests to approve or reject the transfer of registered objects MUST be limited to the registrar that currently sponsors the registered object. Unauthorized attempts to approve or reject the transfer of a registered object MUST be explicitly rejected.

[8] The protocol MUST provide services that allow both the original sponsoring registrar and the potential new registrar to monitor the status of both pending and completed transfer requests.

[9] Transfer of an object MAY extend the object’s registration period. If an object’s registration period will be extended as the result of a transfer, the new expiration date and time MUST be returned after successful completion of a transfer request.

[10] Requests to initiate the transfer of a registered object MUST be available to all authorized registrars.

[11] Registrars might become non-functional and unable to respond to transfer requests. It might be necessary for one registrar to assume management responsibility for the objects associated with another registrar in the event of registrar failure. The protocol MUST NOT restrict the ability to transfer objects in the event of registrar failure.

3.4.6 Object Renewal/Extension

[1] The protocol MUST provide services to renew or extend the validity period of registered domain names. If applicable, the new expiration date and time MUST be returned after successful completion of a request to renew or extend the validity period.

[2] Requests to renew or extend the validity period of a registered object MUST be limited to the registrar that currently sponsors the registered object. Unauthorized attempts to renew or extend the validity period of a registered object MUST be explicitly rejected.

3.4.7 Object Deletion

[1] The protocol MUST provide services to remove a domain name from the registry.

[2] The protocol MUST provide services to remove a name server from the registry.

[3] The protocol MUST provide services to remove a social information object from the registry.

[4] Requests to remove a registered object MUST be limited to the registrar that currently sponsors the registered object. Unauthorized attempts to remove a registered object MUST be explicitly rejected.

3.4.8 Object Existence Query

This section describes requirements for a lightweight query mechanism whose sole purpose is to determine if an object exists in a registry.

[1] The protocol MUST provide services to determine if a domain name exists in the registry. Domain names MUST be searchable by fully qualified name.

[2] The protocol MUST provide services to determine if a name server exists in the registry. Name servers MUST be searchable by fully qualified name.

[3] The protocol MUST provide services to determine if a social information object exists in the registry. Social information MUST be searchable by a registry-unique identifier.

[4] A query to determine if an object exists in the registry MUST return only a positive or negative response so that server software that responds to this query can be optimized for speed.

[5] Requests to determine the existence of a registered object MUST be available to all authorized registrars.

3.4.9 Object Information Query

This section describes requirements for a query mechanism whose purpose is to provide detailed information describing objects that exist in a registry.

[1] The protocol MUST provide services to retrieve information describing a domain name from the registry. Returned information MUST include the identifier of the current sponsoring registrar, the identifier of the registrar that originally registered the domain, the creation date and time, the expiration date and time (if any), the date and time of the last successful update (if any), the identifier of the registrar that performed the last update, the date and time of last completed transfer (if any), the current status of the domain, authorization information, identifiers describing social information associated with the domain, and the subordinate name servers registered in the domain. Authorization information MUST only be returned to the current sponsoring registrar.

[2] The protocol MUST provide services to retrieve information describing a name server from the registry. Returned information MUST include the identifier of the current sponsoring registrar, the identifier of the registrar that originally registered the name server, the creation date and time, the date and time of the last successful update (if any), the identifier of the registrar that performed the last update, the date and time of last completed transfer (if any), and the IP addresses currently associated with the name server.

[3] The protocol MUST provide services to retrieve social information from the registry. Returned information MUST include identification attributes (which MAY include name, address, telephone numbers, and email address), the identifier of the registrar that originally

registered the information, the creation date and time, the date and time of the last successful update (if any), the identifier of the registrar that performed the last update, the date and time of last completed transfer (if any), and authorization information. Authorization information MUST only be returned to the current sponsoring registrar.

[4] The protocol MUST provide services to identify all associated object references, such as name servers associated with domains (including delegations and hierarchical relationships) and contacts associated with domains. This information MUST be visible if the object associations have an impact on the success or failure of protocol operations.

[5] Requests to retrieve information describing a registered object MAY be granted by the registrar that currently sponsors the registered object. Unauthorized attempts to retrieve information describing a registered object MUST be explicitly rejected.

3.5 Domain Status Indicators

[1] The protocol MUST provide status indicators that identify the operational state of a domain name. Indicators MAY be provided to identify a newly created state (the domain has been registered but has not yet appeared in a zone), a normal active state (the domain can be modified and is published in a zone), an inactive state (the domain can be modified but is not published in a zone because it has no authoritative name servers), a hold state (the domain can not be modified and is not published in a zone), a lock state (the domain can not be modified and is published in a zone), a pending transfer state, and a pending removal state.

[2] If provided, protocol indicators for hold and lock status MUST allow independent setting by both registry and registrar.

[3] A domain MAY have multiple statuses at any given time. Some statuses MAY be mutually exclusive.

3.6 Transaction Completion Status

[1] The protocol MUST provide services that unambiguously note the success or failure of every transaction. Individual success and error conditions MUST be noted distinctly.

4. External Interface Requirements

External interfaces define the interaction points between a system and entities that communicate with the system. Specific areas of interest include user interfaces, hardware interfaces, software interfaces, and communications interfaces.

4.1 User, Hardware, and Software Interfaces

[1] The protocol MUST define a wire format for data exchange, not an application design for user, hardware, or software interfaces so that any application able to create the same bits on the wire, and to maintain the image of the same integrity constraints, is a valid implementation of the protocol.

4.2 Communications Interfaces

[1] Registries, registrars, and registrants interact using a wide spectrum of communications interfaces built upon multiple protocols, including transport layer protocols such as TCP and application layer protocols such as SMTP. The protocol MUST only be run over IETF approved protocols that feature congestion control, such as TCP and SCTP.

5. Performance Requirements

[1] Run-time performance is an absolutely critical aspect of protocol usability. While performance is very heavily dependent on the hardware and software architecture that implements a protocol, protocol features can have a direct impact on the ability of the underlying architecture to provide optimal performance. The protocol MUST be usable in both high volume and low volume operating environments.

6. Design Constraints

Protocol designers need to be aware of issues beyond functional and interface requirements when balancing protocol design decisions. This section describes additional factors that might have an impact on protocol design, including standards compliance and hardware limitations.

6.1 Standards Compliance

[1] The protocol MUST conform to current IETF standards. Standards for domain and host name syntax, IP address syntax, security, and transport are particularly relevant. Emerging standards for the Domain Name System MUST be considered as they approach maturity.

[2] The protocol MUST NOT reinvent services offered by lower layer protocol standards. For example, the use of a transport that provides reliability is to be chosen over use of a non-reliable transport with the protocol itself using retransmission to achieve reliability.

6.2 Hardware Limitations

[1] The protocol MUST NOT define any features that preclude hardware independence.

7. Service Attributes

Elements of service beyond functional and interface requirements are essential factors to consider as part of a protocol design effort. This section describes several important service elements to be addressed by protocol designers, including reliability, availability, scalability, maintainability, extensibility, and security.

7.1 Reliability

[1] Reliability is a measure of the extent to which a protocol provides a consistent, dependable level of service. Reliability is an important attribute for a domain name management protocol. An unreliable protocol increases the risk of data exchange errors, which at one extreme can have a direct impact on protocol usability and at the other extreme can introduce discontinuity between registry and registrar data stores. The protocol MUST include features that maximize reliability at the application protocol layer. Services provided by underlying transport, session, and presentation protocols SHOULD also be considered when addressing application protocol reliability.

[2] The protocol MUST be run over the most reliable transport option available in a given environment. The protocol MUST NOT implement a service that is otherwise available in an applicable standard transport.

[3] Default protocol actions for when a request or event times out MUST be well defined.

7.2 Availability

[1] Availability is a measure of the extent to which the services provided by a protocol are accessible for an intended use. Availability of an application layer protocol is primarily dependent on the software and hardware systems that implement the protocol.

The protocol MUST NOT include any features that impinge on the underlying availability of the software and hardware systems needed to service the protocol.

7.3 Scalability

[1] Scalability is a measure of the extent to which a protocol can accommodate use growth while preserving acceptable operational characteristics. The protocol MUST be capable of operating at an acceptable level as the load on registry and registrar systems increases.

7.4 Maintainability

[1] Maintainability is a measure of the extent to which a protocol can be adapted or modified to address unforeseen operational needs or defects. The protocol SHOULD be developed under the nominal working group processes of the IETF to provide a well-known mechanism for ongoing maintenance.

7.5 Extensibility

[1] Extensibility is a measure of the extent to which a protocol can be adapted for future uses that were not readily evident when the protocol was originally designed. The protocol SHOULD provide features that at a minimum allow for the management of new object types without requiring revisions to the protocol itself.

[2] The requirements described in this document are not intended to limit the set of objects that might be managed by the protocol. The protocol MUST include features that allow extension to object types that are not described in this document.

[3] The protocol MUST provide an optional field within all commands whose format and use will be controlled by individual registry policy.

7.6 Security

[1] Transactional privacy and integrity services MUST be available at some protocol layer.

[2] This document describes requirements for basic user identification and authentication services. A generic protocol MAY include additional security services to protect against the attacks described here. A generic protocol MUST depend on other-layered protocols to provide security services that are not provided in the generic protocol itself. A generic protocol that relies on security

services from other-layered protocols MUST specify the protocol layers needed to provide security services.

8. Other Requirements

Certain aspects of anticipated operational environments have to be considered when designing a generic registry-registrar protocol. Areas of concern include database operations, operations, site adaptation, and data collection.

8.1 Database Requirements

[1] The protocol MUST NOT have any database dependencies. However, efficient use of database operations and resources has to be considered as part of the protocol design effort. The protocol SHOULD provide atomic features that can be efficiently implemented to minimize database load.

8.2 Operational Requirements

[1] Registry-registrar interactions at the protocol level SHOULD operate without human intervention. However, intermediate services that preserve the integrity of the protocol MAY be provided. For example, an intermediate service that determines if a registrant is authorized to register a name in a name space can be provided.

[2] The protocol MUST provide services that allow clients and servers to maintain a consistent understanding of the current date and time to effectively manage objects with temporal properties.

8.3 Site Adaptation Requirements

[1] Registries and registrars have varying business and operational requirements. Several factors, including governance standards, local laws, customs, and business practices all play roles in determining how registries and registrars are operated. The protocol MUST be flexible enough to operate in diverse registry-registrar environments.

8.4 Data Collection Requirements

[1] Some of the data exchanged between a registrar and registry might be considered personal, private, or otherwise sensitive. Disclosure of such information might be restricted by laws and/or business practices. The protocol MUST provide services to identify data collection policies.

[2] Some of the social information exchanged between a registrar and registry might be required to create, manage, or operate Internet or DNS infrastructure facilities, such as zone files. Such information is subject to public disclosure per relevant IETF standards.

9. Internationalization Requirements

[1] [RFC1035] describes Internet host and domain names using characters traditionally found in a subset of the 7-bit US-ASCII character set. More recent standards, such as [RFC2130] and [RFC2277], describe the need to develop protocols for an international Internet. These and other standards MUST be considered during the protocol design process to ensure world-wide usability of a generic registry registrar protocol.

[2] The protocol MUST allow exchange of data in formats consistent with current international agreements for the representation of such objects. In particular, this means that addresses MUST include country, that telephone numbers MUST start with the international prefix "+", and that appropriate thought be given to the usability of information in both local and international contexts. This means that some elements (like names and addresses) might need to be represented multiple times, or formatted for different contexts (for instance English/French in Canada, or Latin/ideographic in Japan).

[3] All date and time values specified in a generic registry-registrar protocol MUST be expressed in Universal Coordinated Time. Dates and times MUST include information to represent a four-digit calendar year, a calendar month, a calendar day, hours, minutes, seconds, fractional seconds, and the time zone for Universal Coordinated Time. Calendars apart from the Gregorian calendar MUST NOT be used

10. IANA Considerations

This document does not require any action on the part of IANA. Protocol specifications that require IANA action MUST follow the guidelines described in [RFC2434].

11. Security Considerations

Security services, including confidentiality, authentication, access control, integrity, and non-repudiation SHOULD be applied to protect interactions between registries and registrars as appropriate. Confidentiality services protect sensitive exchanged information from inadvertent disclosure. Authentication services confirm the claimed identity of registries and registrars before engaging in online transactions. Access control services control access to data and

services based on identity. Integrity services guarantee that exchanged data has not been altered between the registry and the registrar. Non-repudiation services provide assurance that the sender of a transaction can not deny being the source of the transaction, and that the recipient cannot deny being the receiver of the transaction.

12. Acknowledgements

This document was originally written as an individual submission Internet-Draft. The provreg working group later adopted it as a working group document and provided many invaluable comments and suggested improvements. The author wishes to acknowledge the efforts of WG chairs Edward Lewis and Jaap Akkerhuis for their process and editorial contributions.

Specific comments that helped guide development of this document were provided by Harald Tveit Alvestrand, Christopher Ambler, Karl Auerbach, Jorg Bauer, George Belotsky, Eric Brunner-Williams, Jordyn Buchanan, Randy Bush, Bruce Campbell, Dan Cohen, Andre Cormier, Kent Crispin, Dave Crocker, Ayesha Damaraju, Lucio De Re, Mats Dufberg, Peter Eisenhauer, Sheer El-Showk, Urs Eppenberger, Patrik Faltstrom, Paul George, Patrick Greenwell, Jarle Greipsland, Olivier Guillard, Alf Hansen, Paul Hoffman, Paul Kane, Shane Kerr, Elmar Knipp, Mike Lampson, Matt Larson, Ping Lu, Klaus Malorny, Bill Manning, Michael Mealling, Patrick Mevzek, Peter Mott, Catherine Murphy, Martin Oldfield, Geva Patz, Elisabeth Porteneuve, Ross Wm. Rader, Budi Rahardjo, Annie Renard, Scott Rose, Takeshi Saigoh, Marcos Sanz, Marcel Schneider, J. William Semich, James Seng, Richard Shockey, Brian Spolarich, William Tan, Stig Venaas, Herbert Vitzthum, and Rick Wesson.

13. References

Normative References:

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

[RFC2434] Narten, T. and H. Alvestrand, "Guidelines for Writing an IANA Considerations Section in RFCs", BCP 26, RFC 2434, October 1998.

Informative References:

[RFC1035] Mockapetris, P., "Domain Names - Implementation and Specification", STD 13, RFC 1035, November 1987.

[RFC2130] Weider, C., Preston, C., Simonsen, K., Alvestrand, H., Atkinson, R., Cripsin, M. and P. Svanberg, "The Report of the IAB Character Set Workshop", RFC 2130, April 1997.

[RFC2277] Alvestrand, H., "IETF Policy on Character Sets and Languages", BCP 18, RFC 2277, January 1998.

14. Editor's Address

Scott Hollenbeck
VeriSign Global Registry Services
21345 Ridgetop Circle
Dulles, VA 20166-6503
USA

EMail: shollenbeck@verisign.com

15. Full Copyright Statement

Acknowledgement

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