Network Working Group
Request for Comments: 4932
Obsoletes: 3732
Category: Standards Track

S. Hollenbeck
VeriSign, Inc.

May 2007

Extensible Provisioning Protocol (EPP) Host Mapping

Status of This Memo

Copyright Notice

Abstract

This document describes an Extensible Provisioning Protocol (EPP) mapping for the provisioning and management of Internet host names stored in a shared central repository. Specified in XML, the mapping defines EPP command syntax and semantics as applied to host names. This document obsoletes RFC 3732.

Table of Contents

1. Introduction

This document describes an Internet host name mapping for version 1.0 of the Extensible Provisioning Protocol (EPP). This mapping is specified using the Extensible Markup Language (XML) 1.0 as described in [W3C.REC-xml-20040204] and XML Schema notation as described in [W3C.REC-xmlschema-1-20041028] and [W3C.REC-xmlschema-2-20041028]. This document obsoletes RFC 3732 [RFC3732].

[RFC4930] provides a complete description of EPP command and response structures. A thorough understanding of the base protocol specification is necessary to understand the mapping described in this document.

XML is case sensitive. Unless stated otherwise, XML specifications and examples provided in this document MUST be interpreted in the character case presented to develop a conforming implementation.

1.1. Relationship of Host Objects and Domain Objects

This document assumes that host name objects have a subordinate relationship to a superordinate domain name object. For example, host name "ns1.example.com" has a subordinate relationship to domain name "example.com". EPP actions (such as object transfers) that do not preserve this relationship MUST be explicitly disallowed.

A host name object can be created in a repository for which no superordinate domain name object exists. For example, host name "ns1.example.com" can be created in the ".example" repository so that DNS domains in ".example" can be delegated to the host. Such hosts are described as "external" hosts in this specification since the name of the host does not belong to the name space of the repository in which the host is being used for delegation purposes.

Whether a host is external or internal relates to the repository in which the host is being used for delegation purposes. Whether or not an internal host is subordinate relates to a domain within the repository. For example, host ns1.example1.com is a subordinate host of domain example1.com, but it is not a subordinate host of domain example2.com. ns1.example1.com can be used as a name server for example2.com. In this case, ns1.example1.com MUST be treated as an internal host, subject to the rules governing operations on subordinate hosts within the same repository.

1.2. 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].

In examples, "C:" represents lines sent by a protocol client and "S:" represents lines returned by a protocol server. Indentation and white space in examples are provided only to illustrate element relationships and are not a REQUIRED feature of this protocol.

2. Object Attributes

An EPP host object has attributes and associated values that can be viewed and modified by the sponsoring client or the server. This section describes each attribute type in detail. The formal syntax for the attribute values described here can be found in the "Formal Syntax" section of this document and in the appropriate normative references.

2.1. Host Names

The syntax for host names described in this document MUST conform to [RFC0952] as updated by [RFC1123]. At the time of this writing, RFC 3490 [RFC3490] describes a standard to use certain ASCII name labels to represent non-ASCII name labels. These conformance requirements might change in the future as a result of progressing work in developing standards for internationalized host names.

2.2. Client Identifiers

All EPP clients are identified by a server-unique identifier. Client identifiers conform to the "clIDType" syntax described in [RFC4930].

2.3. Status Values

A host object MUST always have at least one associated status value. Status values MAY be set only by the client that sponsors a host object and by the server on which the object resides. A client can change the status of a host object using the EPP <update> command. Each status value MAY be accompanied by a string of human-readable text that describes the rationale for the status applied to the object.

A client MUST NOT alter status values set by the server. A server MAY alter or override status values set by a client subject to local server policies. The status of an object MAY change as a result of either a client-initiated transform command or an action performed by a server operator.

Status values that can be added or removed by a client are prefixed with "client". Corresponding status values that can be added or removed by a server are prefixed with "server". Status values that do not begin with either "client" or "server" are server-managed.

Status Value Descriptions:

- clientDeleteProhibited, serverDeleteProhibited

  Requests to delete the object MUST be rejected.

- clientUpdateProhibited, serverUpdateProhibited

  Requests to update the object (other than to remove this status) MUST be rejected.

- linked

  The host object has at least one active association with another object, such as a domain object. Servers SHOULD provide services to determine existing object associations.

- ok

  This is the normal status value for an object that has no pending operations or prohibitions. This value is set and removed by the server as other status values are added or removed.

- pendingCreate, pendingDelete, pendingTransfer, pendingUpdate

  A transform command has been processed for the object (or in the case of a <transfer> command, for the host object's superordinate domain object), but the action has not been completed by the server. Server operators can delay action completion for a variety of reasons, such as to allow for human review or third-party action. A transform command that is processed, but whose requested action is pending, is noted with response code 1001.

When the requested action has been completed, the pendingCreate, pendingDelete, pendingTransfer, or pendingUpdate status value MUST be removed. All clients involved in the transaction MUST be notified using a service message that the action has been completed and that the status of the object has changed.

"ok" status MAY only be combined with "linked" status.

"linked" status MAY be combined with any status.

"pendingDelete" status MUST NOT be combined with either "clientDeleteProhibited" or "serverDeleteProhibited" status.

"pendingUpdate" status MUST NOT be combined with either "clientUpdateProhibited" or "serverUpdateProhibited" status.

The pendingCreate, pendingDelete, pendingTransfer, and pendingUpdate status values MUST NOT be combined with each other.

Other status combinations not expressly prohibited MAY be used.

2.4. Dates and Times

Date and time attribute values MUST be represented in Universal Coordinated Time (UTC) using the Gregorian calendar. The extended date-time form using upper case "T" and "Z" characters defined in [W3C.REC-xmlschema-2-20041028] MUST be used to represent date-time values as XML Schema does not support truncated date-time forms or lower case "T" and "Z" characters.

2.5. IP Addresses

The syntax for IPv4 addresses described in this document MUST conform to [RFC0791]. The syntax for IPv6 addresses described in this document MUST conform to [RFC4291]. Practical considerations for publishing IPv6 address information in zone files are documented in [RFC1886], [RFC2874], and [RFC3152]. A server MAY reject IP addresses that have not been allocated for public use by IANA. When a host object is provisioned for use as a DNS name server, IP addresses SHOULD be required only as needed to generate DNS glue records.

3. EPP Command Mapping

A detailed description of the EPP syntax and semantics can be found in [RFC4930]. The command mappings described here are specifically for use in provisioning and managing Internet host names via EPP.

3.1. EPP Query Commands

EPP provides two commands to retrieve host information: <check> to determine if a host object can be provisioned within a repository, and <info> to retrieve detailed information associated with a host object.

3.1.1. EPP <check> Command

The EPP <check> command is used to determine if an object can be provisioned within a repository. It provides a hint that allows a client to anticipate the success or failure of provisioning an object using the <create> command as object provisioning requirements are ultimately a matter of server policy.

In addition to the standard EPP command elements, the <check> command MUST contain a <host:check> element that identifies the host namespace. The <host:check> element contains the following child elements:

- One or more <host:name> elements that contain the fully qualified names of the host objects to be queried.

Example <check> command:

```
C:<?xml version="1.0" encoding="UTF-8" standalone="no"?>
C:<epp xmlns="urn:ietf:params:xml:ns:epp-1.0">
C:  <command>
C:    <check>
C:      <host:check
C:       xmlns:host="urn:ietf:params:xml:ns:host-1.0">
C:        <host:name>ns1.example.com</host:name>
C:        <host:name>ns2.example.com</host:name>
C:        <host:name>ns3.example.com</host:name>
C:      </host:check>
C:    </check>
C:    <clTRID>ABC-12345</clTRID>
C:  </command>
C:</epp>
```

When a <check> command has been processed successfully, the EPP <resData> element MUST contain a child <host:chkData> element that identifies the host namespace. The <host:chkData> element contains one or more <host:cd> elements that contain the following child elements:

- A <host:name> element that contains the fully qualified name of the queried host object. This element MUST contain an "avail" attribute whose value indicates object availability (can it be provisioned or not) at the moment the <check> command was completed. A value of "1" or "true" means that the object can be provisioned. A value of "0" or "false" means that the object cannot be provisioned.

- An OPTIONAL <host:reason> element that MAY be provided when an object cannot be provisioned. If present, this element contains server-specific text to help explain why the object cannot be provisioned. This text MUST be represented in the response language previously negotiated with the client; an OPTIONAL "lang" attribute MAY be present to identify the language if the negotiated value is something other than the default value of "en" (English).

Example <check> response:

```
S:<?xml version="1.0" encoding="UTF-8" standalone="no"?>
S:<epp xmlns="urn:ietf:params:xml:ns:epp-1.0">
S:  <response>
S:    <result code="1000">
S:      <msg>Command completed successfully</msg>
S:    </result>
S:    <resData>
S:      <host:chkData
S:       xmlns:host="urn:ietf:params:xml:ns:host-1.0">
S:        <host:cd>
S:          <host:name avail="1">ns1.example.com</host:name>
S:        </host:cd>
S:        <host:cd>
S:          <host:name avail="0">ns2.example2.com</host:name>
S:          <host:reason>In use</host:reason>
S:        </host:cd>
S:        <host:cd>
S:          <host:name avail="1">ns3.example3.com</host:name>
S:        </host:cd>
S:      </host:chkData>
S:    </resData>
S:    <trID>
S:      <clTRID>ABC-12345</clTRID>
```

```
S:      <svTRID>54322-XYZ</svTRID>
S:    </trID>
S:  </response>
S:</epp>
```

An EPP error response MUST be returned if a <check> command cannot be processed for any reason.

3.1.2. EPP <info> Command

The EPP <info> command is used to retrieve information associated with a host object. In addition to the standard EPP command elements, the <info> command MUST contain a <host:info> element that identifies the host namespace. The <host:info> element contains the following child elements:

- A <host:name> element that contains the fully qualified name of the host object for which information is requested.

Example <info> command:

```
C:<?xml version="1.0" encoding="UTF-8" standalone="no"?>
C:<epp xmlns="urn:ietf:params:xml:ns:epp-1.0">
C:  <command>
C:    <info>
C:      <host:info
C:       xmlns:host="urn:ietf:params:xml:ns:host-1.0">
C:        <host:name>ns1.example.com</host:name>
C:      </host:info>
C:    </info>
C:    <clTRID>ABC-12345</clTRID>
C:  </command>
C:</epp>
```

When an <info> command has been processed successfully, the EPP <resData> element MUST contain a child <host:infData> element that identifies the host namespace. The <host:infData> element contains the following child elements:

- A <host:name> element that contains the fully qualified name of the host object.

- A <host:roid> element that contains the Repository Object IDentifier assigned to the host object when the object was created.

- One or more <host:status> elements that describe the status of the host object.

- Zero or more <host:addr> elements that contain the IP addresses associated with the host object.

- A <host:clID> element that contains the identifier of the sponsoring client.

- A <host:crID> element that contains the identifier of the client that created the host object.

- A <host:crDate> element that contains the date and time of host object creation.

- A <host:upID> element that contains the identifier of the client that last updated the host object. This element MUST NOT be present if the host object has never been modified.

- A <host:upDate> element that contains the date and time of the most recent host object modification. This element MUST NOT be present if the host object has never been modified.

- A <host:trDate> element that contains the date and time of the most recent successful host object transfer. This element MUST NOT be provided if the host object has never been transferred. Note that host objects MUST NOT be transferred directly; host objects MUST be transferred implicitly when the host object's superordinate domain object is transferred. Host objects that are subject to transfer when transferring a domain object are listed in the response to an EPP <info> command performed on the domain object.

Example <info> response:

```
S:<?xml version="1.0" encoding="UTF-8" standalone="no"?>
S:<epp xmlns="urn:ietf:params:xml:ns:epp-1.0">
S:  <response>
S:    <result code="1000">
S:      <msg>Command completed successfully</msg>
S:    </result>
S:    <resData>
S:      <host:infData
S:       xmlns:host="urn:ietf:params:xml:ns:host-1.0">
S:        <host:name>ns1.example.com</host:name>
S:        <host:roid>NS1_EXAMPLE1-REP</host:roid>
S:        <host:status s="linked"/>
S:        <host:status s="clientUpdateProhibited"/>
```

```
S:        <host:addr ip="v4">192.0.2.2</host:addr>
S:        <host:addr ip="v4">192.0.2.29</host:addr>
S:        <host:addr ip="v6">1080:0:0:0:8:800:200C:417A</host:addr>
S:        <host:clID>ClientY</host:clID>
S:        <host:crID>ClientX</host:crID>
S:        <host:crDate>1999-04-03T22:00:00.0Z</host:crDate>
S:        <host:upID>ClientX</host:upID>
S:        <host:upDate>1999-12-03T09:00:00.0Z</host:upDate>
S:        <host:trDate>2000-04-08T09:00:00.0Z</host:trDate>
S:      </host:infData>
S:    </resData>
S:    <trID>
S:      <clTRID>ABC-12345</clTRID>
S:      <svTRID>54322-XYZ</svTRID>
S:    </trID>
S:  </response>
S:</epp>
```

An EPP error response MUST be returned if an <info> command cannot be processed for any reason.

3.1.3. EPP <transfer> Query Command

Transfer semantics do not directly apply to host objects, so there is no mapping defined for the EPP <transfer> query command.

3.2. EPP Transform Commands

EPP provides three commands to transform host objects: <create> to create an instance of a host object, <delete> to delete an instance of a host object, and <update> to change information associated with a host object. This document does not define host object mappings for the EPP <renew> and <transfer> commands.

Transform commands are typically processed and completed in real time. Server operators MAY receive and process transform commands, but defer completing the requested action if human or third-party review is required before the requested action can be completed. In such situations, the server MUST return a 1001 response code to the client to note that the command has been received and processed, but the requested action is pending. The server MUST also manage the status of the object that is the subject of the command to reflect the initiation and completion of the requested action. Once the action has been completed, all clients involved in the transaction MUST be notified using a service message that the action has been completed and that the status of the object has changed.

3.2.1. EPP <create> Command

The EPP <create> command provides a transform operation that allows a client to create a host object. In addition to the standard EPP command elements, the <create> command MUST contain a <host:create> element that identifies the host namespace. The <host:create> element contains the following child elements:

- A <host:name> element that contains the fully qualified name of the host object to be created.

- Zero or more <host:addr> elements that contain the IP addresses to be associated with the host. Each element MAY contain an "ip" attribute to identify the IP address format. Attribute value "v4" is used to note IPv4 address format. Attribute value "v6" is used to note IPv6 address format. If the "ip" attribute is not specified, "v4" is the default attribute value.

Hosts can be provisioned for use as name servers in the Domain Name System (DNS), described in [RFC1034] and [RFC1035]. Hosts

provisioned as name servers might be subject to server operator policies that require or prohibit specification of IP addresses depending on the name of the host and the name space in which the server will be used as a name server. When provisioned for use as a name server, IP addresses are REQUIRED only as needed to produce DNS glue records. For example, if the server is authoritative for the "com" name space and the name of the server is "ns1.example.net", the server is not required to produce DNS glue records for the name server and IP addresses for the server are not required by the DNS.

If the host name exists in a name space for which the server is authoritative, then the superordinate domain of the host MUST be known to the server before the host object can be created.

Example <create> command:

```
C:<?xml version="1.0" encoding="UTF-8" standalone="no"?>
C:<epp xmlns="urn:ietf:params:xml:ns:epp-1.0">
C:  <command>
C:    <create>
C:      <host:create
C:       xmlns:host="urn:ietf:params:xml:ns:host-1.0">
C:        <host:name>ns1.example.com</host:name>
C:        <host:addr ip="v4">192.0.2.2</host:addr>
C:        <host:addr ip="v4">192.0.2.29</host:addr>
C:        <host:addr ip="v6">1080:0:0:0:8:800:200C:417A</host:addr>
C:      </host:create>
```

```
C:    </create>
C:    <clTRID>ABC-12345</clTRID>
C:  </command>
C:</epp>
```

When a <create> command has been processed successfully, the EPP <resData> element MUST contain a child <host:creData> element that identifies the host namespace. The <host:creData> element contains the following child elements:

- A <host:name> element that contains the fully qualified name of the host object.

- A <host:crDate> element that contains the date and time of host object creation.

Example <create> response:

```
S:<?xml version="1.0" encoding="UTF-8" standalone="no"?>
S:<epp xmlns="urn:ietf:params:xml:ns:epp-1.0">
S:  <response>
S:    <result code="1000">
S:      <msg>Command completed successfully</msg>
S:    </result>
S:    <resData>
S:      <host:creData
S:       xmlns:host="urn:ietf:params:xml:ns:host-1.0">
S:        <host:name>ns1.example.com</host:name>
S:        <host:crDate>1999-04-03T22:00:00.0Z</host:crDate>
S:      </host:creData>
S:    </resData>
S:    <trID>
S:      <clTRID>ABC-12345</clTRID>
S:      <svTRID>54322-XYZ</svTRID>
S:    </trID>
S:  </response>
S:</epp>
```

An EPP error response MUST be returned if a <create> command cannot be processed for any reason.

3.2.2. EPP <delete> Command

The EPP <delete> command provides a transform operation that allows a client to delete a host object. In addition to the standard EPP command elements, the <delete> command MUST contain a <host:delete> element that identifies the host namespace. The <host:delete> element contains the following child elements:

- A <host:name> element that contains the fully qualified name of the host object to be deleted.

A host name object SHOULD NOT be deleted if the host object is associated with any other object. For example, if the host object is associated with a domain object, the host object SHOULD NOT be deleted until the existing association has been broken. Deleting a host object without first breaking existing associations can cause DNS resolution failure for domain objects that refer to the deleted host object.

Example <delete> command:

```
C:<?xml version="1.0" encoding="UTF-8" standalone="no"?>
C:<epp xmlns="urn:ietf:params:xml:ns:epp-1.0">
C:  <command>
C:    <delete>
C:      <host:delete
C:       xmlns:host="urn:ietf:params:xml:ns:host-1.0">
C:        <host:name>ns1.example.com</host:name>
C:      </host:delete>
C:    </delete>
C:    <clTRID>ABC-12345</clTRID>
C:  </command>
C:</epp>
```

When a <delete> command has been processed successfully, a server MUST respond with an EPP response with no <resData> element.

Example <delete> response:

```
S:<?xml version="1.0" encoding="UTF-8" standalone="no"?>
S:<epp xmlns="urn:ietf:params:xml:ns:epp-1.0">
S:  <response>
S:    <result code="1000">
S:      <msg>Command completed successfully</msg>
S:    </result>
S:    <trID>
S:      <clTRID>ABC-12345</clTRID>
S:      <svTRID>54321-XYZ</svTRID>
S:    </trID>
S:  </response>
S:</epp>
```

An EPP error response MUST be returned if a <delete> command cannot be processed for any reason.

3.2.3. EPP <renew> Command

Renewal semantics do not apply to host objects, so there is no mapping defined for the EPP <renew> command.

3.2.4. EPP <transfer> Command

Transfer semantics do not directly apply to host objects, so there is no mapping defined for the EPP <transfer> command. Host objects are subordinate to an existing superordinate domain object, and as such they are subject to transfer when a domain object is transferred.

3.2.5. EPP <update> Command

The EPP <update> command provides a transform operation that allows a client to modify the attributes of a host object. In addition to the standard EPP command elements, the <update> command MUST contain a <host:update> element that identifies the host namespace. The <host:update> element contains the following child elements:

- A <host:name> element that contains the fully qualified name of the host object to be updated.

- An OPTIONAL <host:add> element that contains attribute values to be added to the object.

- An OPTIONAL <host:rem> element that contains attribute values to be removed from the object.

- An OPTIONAL <host:chg> element that contains object attribute values to be changed.

At least one <host:add>, <host:rem>, or <host:chg> element MUST be provided if the command is not being extended. All of these elements MAY be omitted if an <update> extension is present. The <host:add> and <host:rem> elements contain the following child elements:

- One or more <host:addr> elements that contain IP addresses to be associated with or removed from the host object. IP address restrictions described in the <create> command mapping apply here as well.

- One or more <host:status> elements that contain status values to be associated with or removed from the object. When specifying a value to be removed, only the attribute value is significant; element text is not required to match a value for removal.

A <host:chg> element contains the following child elements:

- A <host:name> element that contains a new fully qualified host name by which the host object will be known.

Host name changes MAY require the addition or removal of IP addresses to be accepted by the server. IP address association MAY be subject to server policies for provisioning hosts as name servers.

Host name changes can have an impact on associated objects that refer to the host object. A host name change SHOULD NOT require additional updates of associated objects to preserve existing associations, with one exception: changing an external host object that has associations with objects that are sponsored by a different client. Attempts to update such hosts directly MUST fail with EPP error code 2305. The change can be provisioned by creating a new external host with a new name and needed new attributes and subsequently updating the other objects sponsored by the client.

Example <update> command:

```
C:<?xml version="1.0" encoding="UTF-8" standalone="no"?>
C:<epp xmlns="urn:ietf:params:xml:ns:epp-1.0">
C:  <command>
C:    <update>
C:      <host:update
C:       xmlns:host="urn:ietf:params:xml:ns:host-1.0">
C:        <host:name>ns1.example.com</host:name>
C:        <host:add>
C:          <host:addr ip="v4">192.0.2.22</host:addr>
C:          <host:status s="clientUpdateProhibited"/>
C:        </host:add>
C:        <host:rem>
C:          <host:addr ip="v6">1080:0:0:0:8:800:200C:417A</host:addr>
C:        </host:rem>
C:        <host:chg>
C:          <host:name>ns2.example.com</host:name>
C:        </host:chg>
C:      </host:update>
C:    </update>
C:    <clTRID>ABC-12345</clTRID>
C:  </command>
C:</epp>
```

When an <update> command has been processed successfully, a server MUST respond with an EPP response with no <resData> element.

Example <update> response:

```
S:<?xml version="1.0" encoding="UTF-8" standalone="no"?>
S:<epp xmlns="urn:ietf:params:xml:ns:epp-1.0">
S:  <response>
S:    <result code="1000">
S:      <msg>Command completed successfully</msg>
S:    </result>
S:    <trID>
S:      <clTRID>ABC-12345</clTRID>
S:      <svTRID>54321-XYZ</svTRID>
S:    </trID>
S:  </response>
S:</epp>
```

An EPP error response MUST be returned if an <update> command could not be processed for any reason.

3.3. Offline Review of Requested Actions

Commands are processed by a server in the order they are received from a client. Though an immediate response confirming receipt and processing of the command is produced by the server, a server operator MAY perform an offline review of requested transform commands before completing the requested action. In such situations, the response from the server MUST clearly note that the transform command has been received and processed, but the requested action is pending. The status of the corresponding object MUST clearly reflect processing of the pending action. The server MUST notify the client when offline processing of the action has been completed.

Examples describing a <create> command that requires offline review are included here. Note the result code and message returned in response to the <create> command.

```
S:<?xml version="1.0" encoding="UTF-8" standalone="no"?>
S:<epp xmlns="urn:ietf:params:xml:ns:epp-1.0">
S:  <response>
S:    <result code="1001">
S:      <msg>Command completed successfully; action pending</msg>
S:    </result>
S:    <resData>
S:      <host:creData
S:       xmlns:host="urn:ietf:params:xml:ns:host-1.0">
S:        <host:name>ns1.example.com</host:name>
S:        <host:crDate>1999-04-03T22:00:00.0Z</host:crDate>
S:      </host:creData>
S:    </resData>
```

```
S:    <trID>
S:      <clTRID>ABC-12345</clTRID>
S:      <svTRID>54322-XYZ</svTRID>
S:    </trID>
S:  </response>
S:</epp>
```

The status of the host object after returning this response MUST include "pendingCreate". The server operator reviews the request offline, and informs the client of the outcome of the review either by queuing a service message for retrieval via the <poll> command or by using an out-of-band mechanism to inform the client of the request.

The service message MUST contain text in the <response>, <msgQ>, <msg> element that describes the notification. In addition, the EPP <resData> element MUST contain a child <host:panData> element that identifies the host namespace. The <host:panData> element contains the following child elements:

- A <host:name> element that contains the fully qualified name of the host object. The <host:name> element contains a REQUIRED "paResult" attribute. A positive boolean value indicates that the request has been approved and completed. A negative boolean value indicates that the request has been denied and the requested action has not been taken.

- A <host:paTRID> element that contains the client transaction identifier and server transaction identifier returned with the original response to process the command. The client transaction identifier is OPTIONAL and will only be returned if the client provided an identifier with the original <create> command.

- A <host:paDate> element that contains the date and time describing when review of the requested action was completed.

Example "review completed" service message:

```
S:<?xml version="1.0" encoding="UTF-8" standalone="no"?>
S:<epp xmlns="urn:ietf:params:xml:ns:epp-1.0">
S:  <response>
S:    <result code="1301">
S:      <msg>Command completed successfully; ack to dequeue</msg>
S:    </result>
S:    <msgQ count="5" id="12345">
S:      <qDate>1999-04-04T22:01:00.0Z</qDate>
S:      <msg>Pending action completed successfully.</msg>
S:    </msgQ>
```

```
S:    <resData>
S:      <host:panData
S:       xmlns:host="urn:ietf:params:xml:ns:host-1.0">
S:        <host:name paResult="1">ns1.example.com</host:name>
S:        <host:paTRID>
S:          <clTRID>ABC-12345</clTRID>
S:          <svTRID>54322-XYZ</svTRID>
S:        </host:paTRID>
S:        <host:paDate>1999-04-04T22:00:00.0Z</host:paDate>
S:      </host:panData>
S:    </resData>
S:    <trID>
S:      <clTRID>BCD-23456</clTRID>
S:      <svTRID>65432-WXY</svTRID>
S:    </trID>
S:  </response>
S:</epp>
```

4. Formal Syntax

An EPP object mapping is specified in XML Schema notation. The formal syntax presented here is a complete schema representation of the object mapping suitable for automated validation of EPP XML instances. The BEGIN and END tags are not part of the schema; they are used to note the beginning and ending of the schema for URI registration purposes.

```
BEGIN
<?xml version="1.0" encoding="UTF-8"?>

<schema targetNamespace="urn:ietf:params:xml:ns:host-1.0"
       xmlns:host="urn:ietf:params:xml:ns:host-1.0"
       xmlns:epp="urn:ietf:params:xml:ns:epp-1.0"
       xmlns:eppcom="urn:ietf:params:xml:ns:eppcom-1.0"
       xmlns="http://www.w3.org/2001/XMLSchema"
       elementFormDefault="qualified">

<!--
Import common element types.
-->
 <import namespace="urn:ietf:params:xml:ns:eppcom-1.0"/>
 <import namespace="urn:ietf:params:xml:ns:epp-1.0"/>

 <annotation>
   <documentation>
     Extensible Provisioning Protocol v1.0
     host provisioning schema.
   </documentation>
```

```
 </annotation>

<!--
Child elements found in EPP commands.
-->
 <element name="check" type="host:mNameType"/>
 <element name="create" type="host:createType"/>
 <element name="delete" type="host:sNameType"/>
 <element name="info" type="host:sNameType"/>
 <element name="update" type="host:updateType"/>

<!--
Child elements of the <create> command.
-->
 <complexType name="createType">
   <sequence>
     <element name="name" type="eppcom:labelType"/>
     <element name="addr" type="host:addrType"
      minOccurs="0" maxOccurs="unbounded"/>
   </sequence>
 </complexType>

 <complexType name="addrType">
   <simpleContent>
     <extension base="host:addrStringType">
       <attribute name="ip" type="host:ipType"
        default="v4"/>
     </extension>
   </simpleContent>
 </complexType>

 <simpleType name="addrStringType">
   <restriction base="token">
     <minLength value="3"/>
     <maxLength value="45"/>
   </restriction>
 </simpleType>

 <simpleType name="ipType">
   <restriction base="token">
     <enumeration value="v4"/>
     <enumeration value="v6"/>
   </restriction>
 </simpleType>

<!--
Child elements of the <delete> and <info> commands.
-->
```

```
 <complexType name="sNameType">
   <sequence>
     <element name="name" type="eppcom:labelType"/>
   </sequence>
 </complexType>

<!--
Child element of commands that accept multiple names.
-->
 <complexType name="mNameType">
   <sequence>
     <element name="name" type="eppcom:labelType"
      maxOccurs="unbounded"/>
   </sequence>
 </complexType>
<!--
Child elements of the <update> command.
-->
 <complexType name="updateType">
   <sequence>
     <element name="name" type="eppcom:labelType"/>
     <element name="add" type="host:addRemType"
      minOccurs="0"/>
     <element name="rem" type="host:addRemType"
      minOccurs="0"/>
     <element name="chg" type="host:chgType"
      minOccurs="0"/>
   </sequence>
 </complexType>

<!--
Data elements that can be added or removed.
-->
 <complexType name="addRemType">
   <sequence>
     <element name="addr" type="host:addrType"
      minOccurs="0" maxOccurs="unbounded"/>
     <element name="status" type="host:statusType"
      minOccurs="0" maxOccurs="7"/>
   </sequence>
 </complexType>

<!--
Data elements that can be changed.
-->
 <complexType name="chgType">
   <sequence>
     <element name="name" type="eppcom:labelType"/>
```

```
    </sequence>
  </complexType>

 <!--
 Child response elements.
 -->
  <element name="chkData" type="host:chkDataType"/>
  <element name="creData" type="host:creDataType"/>
  <element name="infData" type="host:infDataType"/>
  <element name="panData" type="host:panDataType"/>

 <!--
 <check> response elements.
 -->
  <complexType name="chkDataType">
    <sequence>
      <element name="cd" type="host:checkType"
       maxOccurs="unbounded"/>
    </sequence>
  </complexType>

  <complexType name="checkType">
    <sequence>
      <element name="name" type="host:checkNameType"/>
      <element name="reason" type="eppcom:reasonType"
       minOccurs="0"/>
    </sequence>
  </complexType>

  <complexType name="checkNameType">
    <simpleContent>
      <extension base="eppcom:labelType">
        <attribute name="avail" type="boolean"
         use="required"/>
      </extension>
    </simpleContent>
  </complexType>

 <!--
 <create> response elements.
 -->
  <complexType name="creDataType">
    <sequence>
      <element name="name" type="eppcom:labelType"/>
      <element name="crDate" type="dateTime"/>
    </sequence>
  </complexType>
```

```
<!--
<info> response elements.
-->
 <complexType name="infDataType">
   <sequence>
     <element name="name" type="eppcom:labelType"/>
     <element name="roid" type="eppcom:roidType"/>
     <element name="status" type="host:statusType"
      maxOccurs="7"/>
     <element name="addr" type="host:addrType"
      minOccurs="0" maxOccurs="unbounded"/>
     <element name="clID" type="eppcom:clIDType"/>
     <element name="crID" type="eppcom:clIDType"/>
     <element name="crDate" type="dateTime"/>
     <element name="upID" type="eppcom:clIDType"
      minOccurs="0"/>
     <element name="upDate" type="dateTime"
      minOccurs="0"/>
     <element name="trDate" type="dateTime"
      minOccurs="0"/>
   </sequence>
 </complexType>

<!--
Status is a combination of attributes and an optional human-readable
message that may be expressed in languages other than English.
-->
 <complexType name="statusType">
   <simpleContent>
     <extension base="normalizedString">
       <attribute name="s" type="host:statusValueType"
        use="required"/>
       <attribute name="lang" type="language"
        default="en"/>
     </extension>
   </simpleContent>
 </complexType>

 <simpleType name="statusValueType">
   <restriction base="token">
     <enumeration value="clientDeleteProhibited"/>
     <enumeration value="clientUpdateProhibited"/>
     <enumeration value="linked"/>
     <enumeration value="ok"/>
     <enumeration value="pendingCreate"/>
     <enumeration value="pendingDelete"/>
     <enumeration value="pendingTransfer"/>
     <enumeration value="pendingUpdate"/>
```

```
      <enumeration value="serverDeleteProhibited"/>
      <enumeration value="serverUpdateProhibited"/>
    </restriction>
  </simpleType>

<!--
Pending action notification response elements.
-->
  <complexType name="panDataType">
    <sequence>
      <element name="name" type="host:paNameType"/>
      <element name="paTRID" type="epp:trIDType"/>
      <element name="paDate" type="dateTime"/>
    </sequence>
  </complexType>
  <complexType name="paNameType">
    <simpleContent>
      <extension base="eppcom:labelType">
        <attribute name="paResult" type="boolean"
         use="required"/>
      </extension>
    </simpleContent>
  </complexType>

<!--
End of schema.
-->
</schema>
END
```

5. Internationalization Considerations

EPP is represented in XML, which provides native support for encoding information using the Unicode character set and its more compact representations including UTF-8. Conformant XML processors recognize both UTF-8 and UTF-16 [RFC2781]. Though XML includes provisions to identify and use other character encodings through use of an "encoding" attribute in an <?xml?> declaration, use of UTF-8 is RECOMMENDED in environments where parser encoding support incompatibility exists.

All date-time values presented via EPP MUST be expressed in Universal Coordinated Time using the Gregorian calendar. XML Schema allows use of time zone identifiers to indicate offsets from the zero meridian, but this option MUST NOT be used with EPP. The extended date-time form using upper case "T" and "Z" characters defined in

[W3C.REC-xmlschema-2-20041028] MUST be used to represent date-time values as XML Schema does not support truncated date-time forms or lower case "T" and "Z" characters.

This document requires host name syntax as specified in [RFC0952] as updated by [RFC1123]. At the time of this writing, RFC 3490 [RFC3490] describes a standard to use certain ASCII name labels to represent non-ASCII name labels. These conformance requirements might change as a result of progressing work in developing standards for internationalized host names.

6. IANA Considerations

This document uses URNs to describe XML namespaces and XML schemas conforming to a registry mechanism described in [RFC3688]. Two URI assignments have been registered by the IANA.

Registration request for the host namespace:

URI: urn:ietf:params:xml:ns:host-1.0

Registrant Contact: See the "Author's Address" section of this document.

XML: None. Namespace URIs do not represent an XML specification.

Registration request for the host XML schema:

URI: urn:ietf:params:xml:schema:host-1.0

Registrant Contact: See the "Author's Address" section of this document.

XML: See the "Formal Syntax" section of this document.

7. Security Considerations

The object mapping described in this document does not provide any security services or introduce any additional considerations beyond those described by [RFC4930] and protocol layers used by EPP.

8. 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 suggestions that have been incorporated into this document were provided by Chris Bason, Jordyn Buchanan, Dave Crocker, Anthony Eden, Sheer El-Showk, Klaus Malorny, Dan Manley, Michael Mealling, Patrick Mevzek, and Rick Wesson.

9. References

9.1. Normative References

[RFC0791] Postel, J., "Internet Protocol", STD 5, RFC 791, September 1981.

[RFC0952] Harrenstien, K., Stahl, M., and E. Feinler, "DoD Internet host table specification", RFC 952, October 1985.

[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.

[RFC1123] Braden, R., "Requirements for Internet Hosts - Application and Support", STD 3, RFC 1123, October 1989.

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

[RFC3688] Mealling, M., "The IETF XML Registry", BCP 81, RFC 3688, January 2004.

[RFC4291] Hinden, R. and S. Deering, "IP Version 6 Addressing Architecture", RFC 4291, February 2006.

[RFC4930] Hollenbeck, S., "Extensible Provisioning Protocol (EPP)", RFC 4930, May 2007.

[W3C.REC-xml-20040204]
Yergeau, F., Maler, E., Sperberg-McQueen, C., Bray, T., and J. Paoli, "Extensible Markup Language (XML) 1.0 (Third Edition)", World Wide Web Consortium FirstEdition REC-xml-20040204, February 2004, <http://www.w3.org/TR/2004/REC-xml-20040204>.

[W3C.REC-xmlschema-1-20041028]
Thompson, H., Maloney, M., Mendelsohn, N., and D. Beech, "XML Schema Part 1: Structures Second Edition", World Wide Web Consortium Recommendation REC-xmlschema-1-20041028, October 2004, <http://www.w3.org/TR/2004/REC-xmlschema-1-20041028>.

[W3C.REC-xmlschema-2-20041028]
Biron, P. and A. Malhotra, "XML Schema Part 2: Datatypes Second Edition", World Wide Web Consortium Recommendation REC-xmlschema-2-20041028, October 2004, <http://www.w3.org/TR/2004/REC-xmlschema-2-20041028>.

9.2. Informative References

[RFC1886] Thomson, S. and C. Huitema, "DNS Extensions to support IP version 6", RFC 1886, December 1995.

[RFC2781] Hoffman, P. and F. Yergeau, "UTF-16, an encoding of ISO 10646", RFC 2781, February 2000.

[RFC2874] Crawford, M. and C. Huitema, "DNS Extensions to Support IPv6 Address Aggregation and Renumbering", RFC 2874, July 2000.

[RFC3152] Bush, R., "Delegation of IP6.ARPA", BCP 49, RFC 3152, August 2001.

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

[RFC3732] Hollenbeck, S., "Extensible Provisioning Protocol (EPP) Host Mapping", RFC 3732, March 2004.

Appendix A. Changes from RFC 3732

1. Minor reformatting as a result of converting I-D source format from nroff to XML.

2. Removed this text from Section 2.3:

   "Transform commands MUST be rejected when a pendingCreate, pendingDelete, pendingTransfer, or pendingUpdate status is set."

3. Changed text in Section 3.2.2 from this:

   "A host name object MUST NOT be deleted if the host object is associated with any other object. For example, if the host object is associated with a domain object, the host object MUST NOT be deleted until the existing association has been broken."

   to this:

   "A host name object SHOULD NOT be deleted if the host object is associated with any other object. For example, if the host object is associated with a domain object, the host object SHOULD NOT be deleted until the existing association has been broken. Deleting a host object without first breaking existing associations can cause DNS resolution failure for domain objects that refer to the deleted host object."

4. Changed text in Section 3.2.5 from "At least one <host:add>, <host:rem>, or <host:chg> element MUST be provided." to "At least one <host:add>, <host:rem>, or <host:chg> element MUST be provided if the command is not being extended. All of these elements MAY be omitted if an <update> extension is present."

5. Changed text in Section 3.3 (old Section 3.2.6) from this:

   "The server operator reviews the request offline, and informs the client of the outcome of the review by queuing a service message for retrieval via the <poll> command."

   to this:

   "The server operator reviews the request offline, and informs the client of the outcome of the review either by queuing a service message for retrieval via the <poll> command or by using an out-of-band mechanism to inform the client of the request."

6. Removed text describing use of the XML Schema schemaLocation attribute. This is an optional attribute that doesn't need to be mandated for use in EPP.

7. Removed references to RFC 3339 and replaced them with references to the W3C XML Schema specification.

8. Replaced references to RFC 3513 with references to RFC 4291.

9. Updated EPP and XML references.

Author's Address

Scott Hollenbeck
VeriSign, Inc.
21345 Ridgetop Circle
Dulles, VA 20166-6503
US

EMail: shollenbeck@verisign.com

Full Copyright Statement

Intellectual Property

Acknowledgement

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