EXTENSIBLE PROVISIONING PROTOCOL MAPPING: <REGISTRY>

Version 1.5.1

Table of Contents

1 Introduction

This document describes a Domain Name Registry Mapping, referred to as Registry Mapping, for the Extensible Provisioning Protocol (EPP) [RFC5730].
A Domain Name Registry can service one or more zones (e.g. top-level domains) with a variety of supported services and policies. This mapping enables the provisioning of the zones in the Domain Name Registry. A Domain Name Registry MAY support a subset of all of the commands defined in this mapping. This mapping is specified using the Extensible Markup Language (XML) 1.0 as described in [X3C.REC-xml-20040204] and XML Schema notation as described in [W3C.REC-xmlschema-1-20041028] and [W3C.REC-xmlschema-2-20041028].

[EPP] 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 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 in examples is provided only to illustrate element relationships and is not a REQUIRED feature of this protocol.

2 Object Attributes

An EPP registry object has attributes and associated values that may 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 Zone Name

The syntax for zone names described in this document MUST conform to RFC 952 [RFC0952] and RFC 1123 [RFC1123]. At the time of this writing, RFC3490 [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 names.

2.2 Phase Values

2.3 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.

2.4 Zone Object

The Zone object, represented by the <registry:zone> element, is the primary object managed by this mapping. The Zone object can apply to any zone level (top level, second level, third level, etc.). The <registry:zone> element contains the following child elements:

- <registry:name> - The zone name that can be at any level (top level, second level, third level, etc.).
- <registry:group> - An OPTIONAL server defined grouping of zones where the zones belong to the same deployable unit.
- <registry:subProduct> - An OPTIONAL sub-product identifier used for the zone and used as the value of the <namestoreExt:subProduct> element of the NameStore Extension.
- <registry:related> - An OPTIONAL definition of the related zones, where there MAY be different forms of relationships. The <registry:related> element contains the following child elements:
    - <registry:fields> - An OPTIONAL definition of how related zone fields are managed. The <registry:fields> element includes a required “type” attribute with either a “shared” or “sync” value. A “shared” value indicates that the server shares a single set of field values across the related zones, where updating a field in one related zone results in updating the field value across all of the related zones. A “sync” value indicates that the field values MUST be synchronized across the related zones by server policy. The <registry:fields> element contains the following child elements:

        - <registry:field> - One or more <registry:field> elements that are shared or MUST be synchronized according to the <registry:fields> "type" attribute value. An example field value is "clID" to refer to the identifier of the sponsoring client or "registrant" to refer to the registrant contact.
    - <registry:zoneMember> - One or more <registry:zoneMember> elements that defines the related zones. The value of the <registry:zoneMember> element is the zone name. The <registry:zoneMember> element includes a required "type" attribute with one of the following values:
        - "primary" – All domain names in the zone MUST be a primary domain name. A primary domain name MUST be the first one created and the last one deleted in the set of related domain names.
        - "alternate" – Domain names of the zone can only be created when the primary domain name exists.
        - "primaryBasedOnCrDate" – A domain name in the zone can be either a primary or alternate domain name based on the earliest created date. The first domain name created in the related zones will become the primary domain name and alternate domain names can be created / deleted.
        - "equal" – There is no concept of primary and alternate domain names, so the related zones are treated as equal. Domain names can be created and deleted in any order.
- <registry:phase> - Zero or more <registry:phase> elements that defines a phase of the zone based on the phases defined in the Launch Phase Mapping for the Extensible Provisioning Protocol (EPP). The "type" attribute defines the phase name / type that include the following possible value with an OPTIONAL "name" attribute that defines the name of the type when the "type" attribute is set to "custom" or the name of the sub-type of the "type". The "mode" attribute reflects the mode in which the phase runs. The <registry:phase> element contains a <registry:startDate> child element that defines the start date and time of the phase and an OPTIONAL <registry:endDate> child element that defines the end date and time of the phase. The "type" attribute supports the following values:
    - "pre-delegation" – Phase when pre-delegation testing is done.
    - "pre-launch" - A phase prior to the sunrise phase where no writable operations will be allowed.
    - "sunrise" - Phase when trademark holders can submit registration applications with trademark information that can be validated by the server.
    - "landrush" – Post sunrise phase when non-trademark holders are allowed to register domain names.
    - "claims" – Trademark claims phase as defined by Trademark Clearinghouse model of displaying a claims notice to clients for domain names that match trademarks.
    - "open" – Post launch phase that is also referred to as "steady state". Servers MAY require additional trademark protection with this phase.
    - "custom" – A custom server launch phase that is defined using the "name" attribute.

  The "mode" attribute supports the following values:
    - "fcfs" – first-come-first-serve. In this mode, each domain name is immediately created and there is no use of an application identifier.
    - "pending-registration' – In this mode, the domain name is created with the pendingCreate status with no use of an application identifier.
    - "pending-application" – In this mode, the domain name, referred to as a domain application, is created in pendingCreate status with the server returning an application identifier in the create response for the client to use in subsequent commands (info,

update, delete). When a domain application is allocated, it will become a domain without the use of an application identifier.

The default mode is "fcfs".

- <registry:services> - The OPTIONAL EPP namespace URIs of the objects and object extensions supported by the server based on RFC 5730. The <registry:services> element contains the following child elements:
  - <registry:objURI> - One or more <registry:objURI> elements that contain namespace URIs representing the objects that the server is capable of managing for the zone with the required "required" attribute that defines whether the server requires the use of object represented by the URI.
  - <registry:svcExtension> - An OPTIONAL element that contains one or more <registry:extURI> elements that contain namespace URIs representing object extensions support by the server for the zone with the required "required" attribute that defines whether the server requires the use of the object extension represented by the URI.
- <registry:slaInfo> - The OPTIONAL Service-Level Agreement (SLA) information for the zone. The SLA information CAN include availability as well as response time SLA's. The <registry:slaInfo> element is a decimal value with the following attributes to define what the SLA value represents:
  - "type" – The type of the SLA
  - "subtype" – The OPTIONAL sub-type of the SLA of the type in the event that there is a hierarchy of SLA types.
  - "command" – The OPTIONAL command that the SLA applies to
  - "unit" – The OPTIONAL unit of the SLA
- <registry:crID> - The OPTIONAL identifier of the client that created the zone.
- <registry:crDate> - The date and time of zone object creation.
- <registry:upID> - The OPTIONAL identifier of the client that last updated the zone object. This element MUST NOT be present if the zone has never been modified.
- <registry:upDate> - The OPTIONAL date and time of the most recent zone object modification. This element MUST NOT be present if the domain object has never been modified.
- <registry:domain> - The domain name object policy information per RFC 5731. The <registry:domain> element contains the following child elements:
  - <registry:domainName> - One or more <registry:domainName> that define the policies for a domain name label for a specific level, defined with the "level" attribute, with a minimum value of "2" for the second level domain name label level. The <registry:domainName> element contains the following child elements:
    - <registry:minLength> - An OPTIONAL minimum length of the domain name label.
    - <registry:maxLength> - An OPTIONAL maximum length of the domain name label.
    - <registry:alphaNumStart> - An OPTIONAL flag indicating whether the label must start with an alphanumeric character with a default of "false".
    - <registry:alphaNumEnd> - An OPTIONAL flag indicating whether the label must end with an alphanumeric character with a default value of "false".
    - <registry:onlyDnsChars> - An OPTIONAL flag indicating whether the label MUST only contain valid DNS characters (alphanumeric and '-') with a default value of "true".

        - <registry:regex> - Zero or more <registry:regex> elements that contain a <registry:expression> child element that defines the regular expression to apply to domain name label along with an OPTIONAL <registry:explanation> child element that describes the regular expression with an OPTIONAL "lang" attribute that defines the language of the explanation with a default value of "en".
        - <registry:reservedNames> - An OPTIONAL element that defines the set of reserved domain names starting from that label level. The reserved names can refer to values with more than one level which is relative to the level of the parent <registry:domainName> element. The <registry:reservedNames> element contains the following child elements:
            - <registry:reservedName> - Zero or more <registry:reservedName> elements containing a reserved domain name relative to the level of the parent <registry:domainName> element.
            - <registry:reservedNameURI> - An OPTIONAL URI that to an externally defined list of reserved domain name relative to the level of the parent <registry:domainName> element.
    - <registry:idn> - The OPTIONAL Internationalized Domain Name (IDN) policy information. The <registry:idn> element contains the following child elements:
        - <registry:idnVersion> - The OPTIONAL server unique version of the IDN language rules.
        - <registry:idnaVersion> - An Internationalizing Domain Names in Applications (IDNA) version supported by the server. IDNA represents a collection of documents that describe the protocol and usage for Internationalized Domain for Applications like IDNA 2003, with value of 2003, or IDNA 2008, with value of 2008.
        - <registry:unicodeVersion> - The Unicode version supported by the server like the value of "6.0" for Unicode 6.0.
        - <registry:encoding> - The OPTIONAL encoding for transforming Unicode characters uniquely and reversibly into DNS compatible characters with a default value of "Punycode".
        - <registry:commingleAllowed> - An OPTIONAL boolean value that indicates whether commingling of scripts is allowed with a default value of "false".
        - <registry:language> - Zero or more <registry:language> elements that defines the supported language codes and character code point policy. The required "code" attribute defines the language code for the supported language. The language code SHOULD be an ISO 639 (ISO 639-1 or ISO 639-2) value. The <registry:language> element contains the following child elements:
            - <registry:table> - The OPTIONAL language table URI that contains the set of code points for the language.
            - <registry:variantStrategy> - An OPTIONAL strategy for the handling of variants for the language. If no <registry:variantStrategy> element is specified then variants are not supported by the language. The possible values for the <registry:variantStrategy> element include:
                - "blocked" – Variant registrations are blocked for all clients
                - "restricted" – Variant registrations are allowed for client of the original IDN registration.
                - "open" – Variant registrations are open to all clients

- <registry:premiumSupport> - The OPTIONAL boolean value that indicates whether the server supports premium domain names with a default value of "false".
- <registry:contactsSupported> - The OPTIONAL boolean value that indicates whether contacts are supported with a default value of "true".
- <registry:contact> - Zero to three <registry:contact> elements that defines the minimum and maximum number of contacts by contact type. The contact type is defined with the required "type" attribute with the possible values of "admin", "tech", and "billing". The <registry:contact> element contains the following child elements:
  - <registry:min> - The minimum number of contacts for the contact type.
  - <registry:max> - The OPTIONAL maximum number of contacts for the contact type. If the <registry:max> element is not defined the maximum number is unbounded.
- <registry:ns> - Defines the minimum and maximum number of delegated host objects (name servers) that can be associated with a domain object. The <registry:ns> element contains the following child elements:
  - <registry:min> - The minimum number of name servers associated with a domain object.
  - <registry:max> - The OPTIONAL maximum number of name servers associated with a domain object. If the <registry:max> element is not defined the maximum number is unbounded.
- <registry:childHost> - Defines the minimum and maximum number of subordinate host objects (child hosts) for a domain object. The <registry: childHost> element contains the following child elements:
  - <registry:min> - The minimum number of child hosts for a domain object.
  - <registry:max> - The OPTIONAL maximum number of child hosts for a domain object. If the <registry:max> element is not defined the maximum number is unbounded.
- <registry:period> - Zero or more <registry:period> elements that defines the supported registration periods and default periods by command type. The required "command" attribute defines the command type with sample values of "create", "renew", and "transfer". The <registry:period> element contains one of the following elements:
  - <registry:length> - The default, minimum, and maximum period length for the command type. The <registry:length> element contains the following child elements, where all of the child elements require the "unit" attribute with possible values of "y" for year and "m" for month:
    - <registry:min> - The minimum supported period length,
    - <registry:max> - The maximum supported period length.
    - <registry:default> - The default period length if not defined by the client.
  - or <registry:serverDecided> - The registration period is decided by the server based on the relationship to a related object that MUST have the same expiration date.
- <registry:transferHoldPeriod> - The period of time a domain object is in the pending transfer before the transfer is auto approved by the server. The <registry:transferHoldPeriod> element MUST have the "unit" attribute with the possible values of "y" for year, "m" for month, and "d" for day.

- <registry:gracePeriod> - Zero or more <registry:gracePeriod> elements that defines the grace periods by operation type. The required “command” attribute defines the operation type with the sample values of “create”, “renew”, “transfer”, and “autoRenew”. The <registry:gracePeriod> element requires the “unit” attribute with the possible values of “d” for day, “h” for hour, and “m” for minute.
- <registry:rgp> - The OPTIONAL Registry Grace Period (RGP) status periods. The <registry:rgp> element contains the following child elements, where each child element supports the “unit” attribute with the possible values of “y” for year, “m” for month, “d” for day, and “h” for hour:
  - <registry:redemptionPeriod> - The length of time that a domain object will remain in the redemptionPeriod status unless the restore request command is received.
  - <registry:pendingRestore> - The length of time that the domain object will remain in the pendingRestore status unless the restore report command is received.
  - <registry:pendingDelete> - The length of time that the domain object will remain in the pendingDelete status prior to be purged.
- <registry:dnssec> - The OPTIONAL DNS Security Extensions (DNSSEC) policies for the server. The <registry:dnssec> element contains the following child elements:
  - <registry:dsDataInterface> - Defines the DS Data Interface, as defined in RFC 5910 [RFC5910], policies. The <registry:dsDataInterface> element contains the following child elements:
    - <registry:min> - The minimum number of DS associated with the domain object.
    - <registry:max> - The maximum number of DS associated with the domain object.
    - <registry:alg> - Zero or more <registry:alg> elements that define the supported algorithms as described in section 5.1.2 of RFC 4034.
    - <registry:digestType> - Zero or more <registry:digestType> elements that define the supported digest types as described in section 5.1.3 of RFC 4034.
  - or <registry:keyDataInterface> - Defines the Key Data Interface, as defined in RFC 5910 [RFC5910], policies. The <registry:keyDataInterface> element contains the following child elements:
    - <registry:min> - The minimum number of keys associated with the domain object.
    - <registry:max> - The maximum number of keys associated with the domain object.
    - <registry:alg> - Zero or more <registry:alg> elements that define the supported algorithms as described in section 2.1.3 of RFC 4034.
  - <registry:maxSigLife> - Defines the maximum signature life policies. The <registry:maxSigLife> element contains the following child elements:
    - <registry:clientDefined> - An OPTIONAL boolean flag indicating whether the client can set the maximum signature life with a default value of “false”.
    - <registry:default> - The OPTIONAL default maximum signature life set by the server.

- <registry:min> - An OPTIONAL minimum signature life supported. The <registry:min> element MUST NOT be defined if the <registry: clientDefined> element value is "false".
- <registry:min> - An OPTIONAL maximum signature life supported. The <registry:max> element MUST NOT be defined if the <registry:clientDefined> element value is "false".

        - <registry:urgent> - An OPTIONAL flag that of whether the client can specify the urgent attribute for DNSSEC updates with a default value of "false".
    - <registry:maxCheckDomain> - The maximum number of domain names (<domain:name> elements) that can be included in a domain check command defined in RFC 5731 [RFC5731].
    - <registry:supportedStatus> - The OPTIONAL set of supported domain statuses defined in RFC 5731 [RFC5731].
    - <registry:authInfoRegEx> - The OPTIONAL regular expression used to validate the domain object authorization information value.
    - <registry:customData> - The OPTIONAL set of custom data using key, value pairs. The <registry:customData> element contains the following child elements:
        - <registry:value> - One or more <registry:value> elements with a required "key" attribute . The "key" attribute defines the key, and the <registry:value> value defined the value of the key, value pair.
- <registry:host> - The host object policy information per RFC 5732. The <registry:host> element contains the following child elements:
    - <registry:internal> - Defines the minimum and maximum number of IP addresses supported for an internal host. The <registry:internal> elements contains the following child elements:
        - <registry:minIP> - Minimum number of IP addresses supported for an internal host.
        - <registry:maxIP> - Maximum number of IP addresses supported for an internal host.
        - <registry:sharePolicy> - The OPTIONAL policy for the sharing of internal hosts in the server. The possible shared policy values include:
            - "perZone" – The internal hosts are shared across all domains of the zone. There is a single pool of internal hosts defined for the zone.
            - "perSystem" – The internal hosts are shared across all zones of the system. There is a single pool of internal hosts across all of the zones supported by the system.
    - <registry:external> - Defines the policies for external hosts. The <registry:external> elements contains the following child elements:
        - <registry:minIP> - Minimum number of IP addresses supported for an external host.
        - <registry:maxIP> - Maximum number of IP addresses supported for an external host.
        - <registry:sharePolicy> - The OPTIONAL policy for the sharing of external hosts in the server. The possible shared policy values include:
            - "perRegistrar" – The external hosts are shared across domains sponsored by an individual registrar client. Each registrar client will have its own set of external hosts to manage.

- "perZone" – The external hosts are shared across all domains of the zone. There is a single pool of external hosts defined for the zone.
- "perSystem" – The external hosts are shared across all zones of the system. There is a single pool of external hosts across all of the zones supported by the system.

- <registry:nameRegex> - Zero or more <registry:nameRegex> elements that define the regular expressions used to validate the host name value.
- <registry:maxCheckHost> - The maximum number of host names (<domain:name> elements) that can be included in a host check command defined in RFC 5732 [RFC5732].
- <registry:supportedStatus> - The OPTIONAL set of supported host statuses defined in RFC 5732 [RFC5732].
- <registry:customData> - The OPTIONAL set of custom data using key, value pairs. The <registry:customData> element contains the following child elements:
  - <registry:value> - One or more <registry:value> elements with a required "key" attribute . The "key" attribute defines the key, and the <registry:value> value defined the value of the key, value pair.

- <registry:contact> - The OPTIONAL contact object policy information per RFC 5733. The <registry:contact> element contains the following child elements:
  - <registry:contactIdRegEx> - The OPTIONAL regular expression used to validate the <contact:id> element defined in RFC 5733 [RFC5733].
  - <registry:sharePolicy> - The OPTIONAL policy for the sharing of contacts in the server. The possible shared policy values include:
    - "perZone" – The contacts are shared across all objects of the zone. There is a single pool of contacts defined for the zone.
    - "perSystem" – The contacts are shared across all zones of the system. There is a single pool of contacts across all of the zones supported by the system.
  - <registry:intSupport> - A boolean value that defines whether the server supports the internationalized form of postal-address information using the type="int" attribute of RFC 5733 [RFC5733].
  - <registry:locSupport> - A boolean value that defines whether the server supports the localized form of postal-address information using the type="loc" attribute of RFC 5733 [RFC5733].
  - <registry:postalInfo> - The postal-address information policy information. The <registry:postalInfo> element contains the following child elements:
    - <registry:name> - The minimum and maximum length of <contact:name> element defined RFC 5733 [RFC5733] using the <registry:minLength> and <registry:maxLength> child elements, respectively.
    - <registry:org> - The minimum and maximum length of the <contact:org> element defined in RFC 5733 [RFC5733] using the <registry:minLength> and <registry:maxLength> child elements, respectively.
    - <registry:address> - The address information policy information. The <registry:address> element contains the following child elements:
      - <registry:street> - The minimum and maximum length and the minimum and maximum number of the <contact:street> elements defined in RFC 5733 [RFC5733]. The <registry:street> element contains the following child elements:

- <registry:minLength> - The minimum length of the <contact:street> elements.
- <registry:maxLength> - The maximum length of the <contact:street> elements.
- <registry:minEntry> - The minimum number of <contact:street> elements.
- <registry:maxEntry> - The maximum number of <contact:street> elements.

- <registry:city> - The minimum and maximum length of the <contact:city> element defined in RFC 5733 [RFC5733] using the <registry:minLength> and <registry:maxLength> child elements, respectively.
- <registry:sp> - The minimum and maximum length of the <contact:sp> element defined in RFC 5733 [RFC5733] using the <registry:minLength> and <registry:maxLength> child elements, respectively.
- <registry:pc> - The minimum and maximum length of the <contact:pc> element defined in RFC 5733 [RFC5733] using the <registry:minLength> and <registry:maxLength> child elements, respectively.

- <registry:voiceRequired> - An OPTIONAL boolean flag indicating whether the server requires the <contact:voice> element to be defined, with a default value of "false".
- <registry:voiceExt> - The OPTIONAL minimum and maximum length of the <contact:voice> extension "x" attribute defined in RFC 5733 [RFC5733] using the <registry:minLength> and <registry:maxLength> child elements, respectively.
- <registry:faxExt> - The OPTIONAL minimum and maximum length of the <contact:fax> extension "x" attribute defined in RFC 5733 [RFC5733] using the <registry:minLength> and <registry:maxLength> child elements, respectively.
- <registry:emailRegex> - Zero or more <registry:emailRegex> elements that define the regular expressions used to validate the <contact:email> element defined in RFC 5733 [RFC5733].

- <registry:maxCheckContact> - The maximum number of contact identifiers (<contact:id> elements) that can be included in a contact check command defined in RFC 5733 [RFC5733].
- <registry:authInfoRegex> - The OPTIONAL regular expression used to validate the contact object authorization information value.
- <registry:clientDisclosureSupported> - The OPTIONAL flag that indicates whether the server supports the client to identify elements that require exception server-operator handling to allow or restrict disclosure to third parties defined in RFC 5733 [RFC5733] with a default of "false".
- <registry:supportedStatus> - The OPTIONAL set of supported contact statuses defined in RFC 5733 [RFC5733].
- <registry:transferHoldPeriod> - The OPTIONAL period of time a contact object is in the pending transfer before the transfer is auto approved by the server. The

<registry:transferHoldPeriod> element MUST have the “unit” attribute with the possible values of “y” for year, “m” for month, and “d” for day.

- <registry:customData> - The OPTIONAL set of custom data using key, value pairs. The <registry:customData> element contains the following child elements:
  - <registry:value> - One or more <registry:value> elements with a required “key” attribute . The “key” attribute defines the key, and the <registry:value> value defined the value of the key, value pair.

Example of a <registry:zone> element:

```
<registry:zone>
        <registry:name>EXAMPLE</registry:name>
        <registry:group>STANDARD</registry:group>
        <registry:subProduct>EXAMPLE</registry:subProduct>
        <registry:related>
                <registry:fields type="sync">
                  <registry:field>clID</registry:field>
                  <registry:field>registrant</registry:field>
                  <registry:field>ns</registry:field>
                </registry:fields>
                <registry:zoneMember type="equal">EXAMPLE</registry:zoneMember>
                <registry:zoneMember type="equal">EXAMPLE2</registry:zoneMember>
                <registry:zoneMember type="equal">EXAMPLE3</registry:zoneMember>
        </registry:related>
        <registry:phase type="sunrise">
                <registry:startDate>2012-11-01T00:00:00.0Z</registry:startDate>
                <registry:endDate>2012-12-01T00:00:00.0Z</registry:endDate>
        </registry:phase>
        <registry:phase type="claims" name="landrush">
                <registry:startDate>2012-12-01T00:00:00.0Z</registry:startDate>
                <registry:endDate>2012-12-08T00:00:00.0Z</registry:endDate>
        </registry:phase>
        <registry:phase type="claims" name="open">
                <registry:startDate>2012-12-08T00:00:00.0Z</registry:startDate>
                <registry:endDate>2013-02-01T00:00:00.0Z</registry:endDate>
        </registry:phase>
        <registry:phase type="open">
                <registry:startDate>2013-02-01T00:00:00.0Z</registry:startDate>
        </registry:phase>
        <registry:services>
                <registry:objURI required="true">urn:ietf:params:xml:ns:domain-1.0
                </registry:objURI>
                <registry:objURI required="true">urn:ietf:params:xml:ns:host-1.0
                </registry:objURI>
                <registry:objURI required="true">urn:ietf:params:xml:ns:contact-1.0
                </registry:objURI>
                <registry:svcExtension>
                  <registry:extURI required="true">urn:ietf:params:xml:ns:rgp-1.0
                  </registry:extURI>
```

```
            <registry:extURI required="true">urn:ietf:params:xml:ns:secDNS-1.1
            </registry:extURI>
            <registry:extURI required="true">http://www.verisign-grs.com/epp/namestoreExt-
1.1
            </registry:extURI>
            <registry:extURI required="false">http://www.verisign.com/epp/idnLang-1.0
            </registry:extURI>
          </registry:svcExtension>
    </registry:services>
    <registry:slaInfo>
          <registry:sla type="downtime" unit="min">
           864
          </registry:sla>
          <registry:sla type="rtt" command="domain:check" unit="ms">
           2000
          </registry:sla>
          <registry:sla type="rtt" command="domain:info" unit="ms">
           2000
          </registry:sla>
          <registry:sla type="rtt" command="domain:create" unit="ms">
           4000
          </registry:sla>
          <registry:sla type="rtt" command="domain:update" unit="ms">
           4000
          </registry:sla>
          <registry:sla type="rtt" command="domain:renew" unit="ms">
           4000
          </registry:sla>
          <registry:sla type="rtt" command="domain:delete" unit="ms">
           4000
          </registry:sla>
          <registry:sla type="rtt" command="domain:transfer" unit="ms">
           4000
          </registry:sla>
    </registry:slaInfo>
    <registry:crID>clientX</registry:crID>
    <registry:crDate>2012-10-01T00:00:00.0Z</registry:crDate>
    <registry:upID>clientY</registry:upID>
    <registry:upDate>2012-10-15T00:00:00.0Z</registry:upDate>
    <registry:domain>
          <registry:domainName level="2">
           <registry:minLength>5</registry:minLength>
           <registry:maxLength>50</registry:maxLength>
           <registry:alphaNumStart>true</registry:alphaNumStart>
           <registry:alphaNumEnd>false</registry:alphaNumEnd>
           <registry:onlyDnsChars>true</registry:onlyDnsChars>
           <registry:regex>
                <registry:expression>^\w+.*$</registry:expression>
                <registry:explanation>Alphanumeric</registry:explanation>
```

```
  </registry:regex>
  <registry:regex>
        <registry:expression>^\d+.*$</registry:expression>
  </registry:regex>
  <registry:reservedNames>
        <registry:reservedName>reserved1</registry:reservedName>
  </registry:reservedNames>
</registry:domainName>
<registry:idn>
  <registry:idnVersion>4.1</registry:idnVersion>
  <registry:idnaVersion>2008</registry:idnaVersion>
  <registry:unicodeVersion>6.0</registry:unicodeVersion>
  <registry:encoding>Punycode</registry:encoding>
  <registry:commingleAllowed>false</registry:commingleAllowed>
  <registry:language code="LANG-1">
        <registry:table>http://www.iana.org/idn-tables/test_tab1_1.1.txt
        </registry:table>
        <registry:variantStrategy>blocked</registry:variantStrategy>
  </registry:language>
</registry:idn>
<registry:premiumSupport>false</registry:premiumSupport>
<registry:contact type="admin">
  <registry:min>1</registry:min>
  <registry:max>4</registry:max>
</registry:contact>
<registry:ns>
  <registry:min>0</registry:min>
  <registry:max>13</registry:max>
</registry:ns>
<registry:childHost>
  <registry:min>0</registry:min>
</registry:childHost>
<registry:period command="create">
  <registry:length>
        <registry:min unit="y">1</registry:min>
        <registry:max unit="y">10</registry:max>
        <registry:default unit="y">1</registry:default>
  </registry:length>
</registry:period>
<registry:transferHoldPeriod unit="d">5
</registry:transferHoldPeriod>
<registry:gracePeriod command="create" unit="d">5
</registry:gracePeriod>
<registry:gracePeriod command="renew" unit="d">5
</registry:gracePeriod>
<registry:gracePeriod command="transfer" unit="d">5
</registry:gracePeriod>
<registry:gracePeriod command="autoRenew" unit="d">45
</registry:gracePeriod>
```

```
        <registry:rgp>
          <registry:redemptionPeriod unit="d">30
          </registry:redemptionPeriod>
          <registry:pendingRestore unit="d">7
          </registry:pendingRestore>
          <registry:pendingDelete unit="d">5
          </registry:pendingDelete>
        </registry:rgp>
        <registry:dnssec>
          <registry:dsDataInterface>
                <registry:min>0</registry:min>
                <registry:max>13</registry:max>
                <registry:alg>3</registry:alg>
                <registry:digestType>1</registry:digestType>
          </registry:dsDataInterface>
          <registry:maxSigLife>
                <registry:clientDefined>false</registry:clientDefined>
          </registry:maxSigLife>
        </registry:dnssec>
        <registry:maxCheckDomain>5</registry:maxCheckDomain>
        <registry:supportedStatus>
          <registry:status>ok</registry:status>
          <registry:status>clientDeleteProhibited</registry:status>
          <registry:status>serverDeleteProhibited</registry:status>
          <registry:status>clientHold</registry:status>
          <registry:status>serverHold</registry:status>
          <registry:status>clientRenewProhibited</registry:status>
          <registry:status>serverRenewProhibited</registry:status>
          <registry:status>clientTransferProhibited</registry:status>
          <registry:status>serverTransferProhibited</registry:status>
          <registry:status>clientUpdateProhibited</registry:status>
          <registry:status>serverUpdateProhibited</registry:status>
          <registry:status>inactive</registry:status>
          <registry:status>pendingDelete</registry:status>
          <registry:status>pendingTransfer</registry:status>
        </registry:supportedStatus>
        <registry:authInfoRegex>
          <registry:expression>^.*$</registry:expression>
        </registry:authInfoRegex>
  </registry:domain>
  <registry:host>
        <registry:internal>
          <registry:minIP>1</registry:minIP>
          <registry:maxIP>13</registry:maxIP>
          <registry:sharePolicy>perZone</registry:sharePolicy>
        </registry:internal>
        <registry:external>
          <registry:minIP>0</registry:minIP>
          <registry:maxIP>0</registry:maxIP>
```

```
            <registry:sharePolicy>perZone</registry:sharePolicy>
          </registry:external>
          <registry:nameRegex>
            <registry:expression>^.*$</registry:expression>
          </registry:nameRegex>
          <registry:maxCheckHost>5</registry:maxCheckHost>
          <registry:supportedStatus>
            <registry:status>ok</registry:status>
            <registry:status>clientDeleteProhibited</registry:status>
            <registry:status>serverDeleteProhibited</registry:status>
            <registry:status>clientUpdateProhibited</registry:status>
            <registry:status>serverUpdateProhibited</registry:status>
            <registry:status>linked</registry:status>
            <registry:status>pendingDelete</registry:status>
            <registry:status>pendingTransfer</registry:status>
          </registry:supportedStatus>
    </registry:host>
    <registry:contact>
     <registry:contactIdRegex>
            <registry:expression>^.*$</registry:expression>
          </registry:contactIdRegex>
          <registry:sharePolicy>perZone</registry:sharePolicy>
          <registry:intSupport>true</registry:intSupport>
          <registry:locSupport>false</registry:locSupport>
          <registry:postalInfo>
            <registry:name>
                  <registry:minLength>5</registry:minLength>
                  <registry:maxLength>15</registry:maxLength>
            </registry:name>
            <registry:org>
                  <registry:minLength>2</registry:minLength>
                  <registry:maxLength>40</registry:maxLength>
            </registry:org>
            <registry:address>
                  <registry:street>
                        <registry:minLength>1</registry:minLength>
                        <registry:maxLength>40</registry:maxLength>
                        <registry:minEntry>1</registry:minEntry>
                        <registry:maxEntry>3</registry:maxEntry>
                  </registry:street>
                  <registry:city>
                        <registry:minLength>1</registry:minLength>
                        <registry:maxLength>40</registry:maxLength>
                  </registry:city>
                  <registry:sp>
                        <registry:minLength>1</registry:minLength>
                        <registry:maxLength>40</registry:maxLength>
                  </registry:sp>
                  <registry:pc>
```

```
                        <registry:minLength>1</registry:minLength>
                        <registry:maxLength>40</registry:maxLength>
                    </registry:pc>
                </registry:address>
                <registry:voiceRequired>false</registry:voiceRequired>
                <registry:voiceExt>
                    <registry:minLength>1</registry:minLength>
                    <registry:maxLength>40</registry:maxLength>
                </registry:voiceExt>
                <registry:faxExt>
                    <registry:minLength>1</registry:minLength>
                    <registry:maxLength>40</registry:maxLength>
                </registry:faxExt>
                <registry:emailRegex>
                    <registry:expression>^.+\..+$</registry:expression>
                </registry:emailRegex>
            </registry:postalInfo>
            <registry:maxCheckContact>5</registry:maxCheckContact>
            <registry:authInfoRegex>
              <registry:expression>^.*$</registry:expression>
            </registry:authInfoRegex>
            <registry:clientDisclosureSupported>false
            </registry:clientDisclosureSupported>
            <registry:supportedStatus>
              <registry:status>ok</registry:status>
              <registry:status>clientDeleteProhibited</registry:status>
              <registry:status>serverDeleteProhibited</registry:status>
              <registry:status>clientTransferProhibited</registry:status>
              <registry:status>serverTransferProhibited</registry:status>
              <registry:status>clientUpdateProhibited</registry:status>
              <registry:status>serverUpdateProhibited</registry:status>
              <registry:status>linked</registry:status>
              <registry:status>pendingDelete</registry:status>
              <registry:status>pendingTransfer</registry:status>
            </registry:supportedStatus>
            <registry:transferHoldPeriod unit="d">5
            </registry:transferHoldPeriod>
        </registry:contact>
</registry:zone>
```

3 EPP Command Mapping

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

3.1 EPP Query Commands

EPP provides three commands to retrieve object information: <check> to determine if an object is available for provisioning, <info> to retrieve detailed information associated with an object, and <transfer> to retrieve object transfer status information.

3.1.1 EPP <check> Command

The EPP <check> command is used to determine if the server currently supports a zone. If the response indicates that the zone is not available, then it is currently supported; otherwise it MAY be available to be created by an authorized client.

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

- One or more <registry:name> elements that contain the fully qualified names of the zone 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:      <registry:check
C:       xmlns:registry ="http://www.verisign.com/epp/registry-1.0">
C:        <registry:name>zone1</registry:name>
C:        <registry:name>zone2</registry:name>
C:        <registry:name>zone3</registry:name>
C:      </registry: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 <registry:chkData> element that identifies the registry namespace. The <registry:chkData> element contains one or more <registry:cd> elements that contain the following child elements:

- A <registry:name> element that contains the fully qualified name of the queried zone object. This element MUST contain an "avail" attribute whose value indicates zone is currently supported or availability at the moment the <check> command was completed for an

authorized client. A value of "1" or "true" means that the zone object is available for an authorized client. A value of "0" or "false" means that the zone object is currently supported by the server.

- An OPTIONAL <registry:reason> element that MAY be provided when a zone object is not available for provisioning. If present, this element contains server-specific text to help explain why the zone object is unavailable. 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 a 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:      <registry:chkData
S:       xmlns:registry="http://www.verisign.com/epp/registry-1.0 ">
S:        <registry:cd>
S:          <registry:name avail="0">zone1</registry:name>
S:          <registry:reason>Client not authorized</registry:reason>
S:        </registry:cd>
S:        <registry:cd>
S:          <registry:name avail="0">zone2</registry:name>
S:          <registry:reason>Already supported</registry:reason>
S:        </registry:cd>
S:        <registry:cd>
S:          <registry:name avail="1">zone3</registry:name>
S:        </registry:cd>
S:      </ registry: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 zone object. The response to this command MAY vary depending on the identity of the querying client, use of authorization information, and server policy towards unauthorized clients. Server policy determines which OPTIONAL elements are returned.

In addition to the standard EPP command elements, the <info> command MUST contain a <registry:info> element that identifies the registry namespace. The <registry:info> element contains the following one of the two child elements:

- A <registry:all> element that is empty and that indicates that a list of all of the supported zone objects are queried with a summary set of attributes per zone object.
- or a <registry:name> element that contains the fully qualified name of the zone object to be queried for a full set of attributes for the zone object.

Example <info> command to query for a summary set of attributes for all of the supported zone objects:

```
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:      <registry:info
C:       xmlns:registry="http://www.verisign.com/epp/registry-1.0">
C:        <registry:all/>
C:      </registry:info>
C:    </info>
C:    <clTRID>ABC-12345</clTRID>
C:  </command>
C:</epp>
```

Example <info> command to query for the full set of “zone1” zone object attributes:

```
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:      <registry:info
C:       xmlns:registry="http://www.verisign.com/epp/registry-1.0">
C:        <registry:name>zone1</registry:name>
C:      </registry: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 <registry:infData> element that identifies the registry namespace. The <registry:infData> element contains one of the two following child elements:

- A <registry:zoneList> element that contains the list of supported zones by the server with a set of summary attributes per zone. The <registry:zoneList> element contains the following child elements:
    -
- Or a <registry:zone> element that contains the full set of attributes for the zone name as defined in the section 2.4.

Example <info> response to a query for a summary of all of the supported zone objects:

```
S:<?xml version="1.0" encoding="UTF-8" standalone="no"?>
S:<epp xmlns="urn:ietf:params:xml:ns:epp-1.0"
S:    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
S:  <response>
S:   <result code="1000">
S:     <msg>Command completed successfully</msg>
S:   </result>
S:   <resData>
S:     <registry:infData
S:      xmlns:registry="http://www.verisign.com/epp/registry-1.0">
S:          <registry:zoneList>
S:         <registry:zone>
S:           <registry:name>EXAMPLE1</registry:name>
S:                    <registry:crDate>2012-10-01T00:00:00.0Z</registry:crDate>
S:           <registry:upDate>2012-10-15T00:00:00.0Z</registry:upDate>
S:           </registry:zone>
S:         <registry:zone>
S:           <registry:name>EXAMPLE2</registry:name>
S:                    <registry:crDate>2012-09-01T00:00:00.0Z</registry:crDate>
S:                    <registry:upDate>2012-09-19T00:00:00.0Z</registry:upDate>
S:            </registry:zone>
S:          </registry:zoneList>
S:     </registry:infData>
S:   </resData>
S:   <trID>
S:     <clTRID>ABC-12345</clTRID>
S:     <svTRID>54322-XYZ</svTRID>
S:   </trID>
S:  </response>
S:</epp>
```

Example <info> response to query for the full set of “EXAMPLE” zone object attributes:

```
S:<?xml version="1.0" encoding="UTF-8" standalone="no"?>
```

```
S:<epp xmlns="urn:ietf:params:xml:ns:epp-1.0"
S:    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
S:  <response>
S:    <result code="1000">
S:      <msg>Command completed successfully</msg>
S:    </result>
S:    <resData>
S:      <registry:infData
S:       xmlns:registry="http://www.verisign.com/epp/registry-1.0">
S:         <registry:zone>
S:           <registry:name>EXAMPLE</registry:name>
S:            …
S:        </registry:zone>
S:      </registry: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> Command

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

3.2 EPP Transform Commands

EPP provides five commands to transform zone objects: <create> to create an instance of a zone object, <delete> to delete an instance of a zone object, and <update> to change information associated with a zone object.

3.2.1 EPP <create> Command

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

- Or a <registry:zone> element that contains the full set of attributes for the zone to create as defined in the section 2.4.

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:     xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
C:  <command>
C:    <create>
C:      <registry:create
C:       xmlns:registry="http://www.verisign.com/epp/registry-1.0">
C:         <registry:zone>
C:           <registry:name>EXAMPLE</registry:name>
C:             …
C:       </registry:zone>
C:      </registry: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 <registry:creData> element that identifies the registry namespace. The <registry:creData> element contains the following child elements:

A <registry:name> element that contains the fully qualified name of the zone object
A <registry:crDate> element that contains the date and time of zone 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:     xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
S:  <response>
S:    <result code="1000">
```

```
S:      <msg>Command completed successfully</msg>
S:    </result>
S:    <resData>
S:      <registry:creData
S:       xmlns:registry="http://www.verisign.com/epp/registry-1.0">
S:        <registry:name>zone1</registry:name>
S:        <registry:crDate>2012-10-30T22:00:00.0Z</registry:crDate>
S:      </registry:creData>
S:    </resData>
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 <create> command can not 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 zone object. In addition to the standard EPP command elements, the <delete> command MUST contain a <registry:delete> element that identifies the registry namespace. The <registry:delete> element contains the following child elements:

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

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:      <registry:delete
C:       xmlns: registry="http://www.verisign.com/epp/registry-1.0">
C:        <registry:name>EXAMPLE</registry:name>
C:      </registry:delete>
C:    </delete>
C:    <clTRID>ABC-12345</clTRID>
C:  </command>
C:</epp>
```

When a <delete> zone 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 can not be processed for any reason.

3.2.3 EPP <renew> Command

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

3.2.5 EPP <update> Command

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

- Or a <registry:zone> element that contains the full set of attributes for the zone to that will be set as defined in the section 2.4. The update completely replaces the prior version of the zone.

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:     xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
C:  <command>
C:    <update>
C:      <registry:update
C:       xmlns:registry="http://www.verisign.com/epp/registry-1.0">
C:         <registry:zone>
C:           <registry:name>EXAMPLE</registry:name>
C:             …
C:         </registry:zone>
C:      </registry:update>
C:    </create>
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:     xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
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 can not be processed for any reason.

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 xmlns:registry="http://www.verisign.com/epp/registry-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"
  targetNamespace="http://www.verisign.com/epp/registry-1.0"
  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
       registry provisioning schema.
     </documentation>
  </annotation>
  <!--
  Child elements found in EPP commands.
  -->
  <element name="check" type="registry:mNameType"/>
  <element name="create" type="registry:createType"/>
  <element name="delete" type="registry:sNameType"/>
  <element name="info" type="registry:infoType"/>
  <element name="update" type="registry:updateType"/>
  <!--
    Child elements of the <check> command.
  -->
  <complexType name="mNameType">
    <sequence>
      <element name="name" type="eppcom:labelType"
      maxOccurs="unbounded"/>
    </sequence>
  </complexType>
  <!--
    Child elements of the <delete> command.
  -->
  <complexType name="sNameType">
    <sequence>
      <element name="name" type="eppcom:labelType"/>
    </sequence>
  </complexType>
  <!--
    Child elements of the <create> command.
  -->
  <complexType name="createType">
```

```
  <sequence>
    <element name="zone" type="registry:zoneType"/>
  </sequence>
</complexType>
<complexType name="updateType">
  <sequence>
    <element name="zone" type="registry:zoneType"/>
  </sequence>
</complexType>
<!--
  Child elements of the <info> command.
-->
<complexType name="infoType">
  <sequence>
    <choice>
      <element name="all">
        <complexType/>
      </element>
      <element name="name" type="eppcom:labelType"/>
    </choice>
  </sequence>
</complexType>

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

<!--
  <create> response elements.
-->
<complexType name="creDataType">
  <sequence>
    <element name="name" type="eppcom:labelType"/>
    <element name="crDate" type="dateTime"/>
  </sequence>
</complexType>
<!--
  <check> response elements.
-->
<complexType name="chkDataType">
  <sequence>
    <element name="cd" type="registry:checkType"
    maxOccurs="unbounded"/>
  </sequence>
</complexType>
<complexType name="checkType">
  <sequence>
    <element name="name" type="registry: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>
<!--
  <info> response elements.
-->
<complexType name="infDataType">
  <choice>
    <element name="zoneList" type="registry:zoneListType"/>
    <element name="zone" type="registry:zoneType"/>
  </choice>
</complexType>
<complexType name="zoneListType">
  <sequence>
    <element name="zone" type="registry:zoneSummaryType"
    minOccurs="0" maxOccurs="unbounded"/>
  </sequence>
</complexType>
<complexType name="zoneSummaryType">
  <sequence>
    <element name="name" type="string"/>
    <element name="crDate" type="dateTime"/>
    <element name="upDate" type="dateTime"
    minOccurs="0"/>
  </sequence>
</complexType>
<complexType name="zoneType">
  <sequence>
    <element name="name" type="eppcom:labelType"/>
    <element name="group" type="token"
    minOccurs="0"/>
    <element name="subProduct" type="token"
    minOccurs="0"/>
    <element name="related" type="registry:relatedType"
    minOccurs="0"/>
    <element name="phase" type="registry:phaseType"
    minOccurs="0" maxOccurs="unbounded"/>
    <element name="services" type="registry:servicesType"
    minOccurs="0"/>
    <element name="slaInfo" type="registry:slaInfoType"
    minOccurs="0"/>
    <element name="crID" type="eppcom:clIDType"
    minOccurs="0"/>
    <element name="crDate" type="dateTime"/>
    <element name="upID" type="eppcom:clIDType"
    minOccurs="0"/>
    <element name="upDate" type="dateTime"
    minOccurs="0"/>
    <element name="domain" type="registry:domainType"/>
    <element name="host" type="registry:hostType"/>
    <element name="contact" type="registry:contactType"
    minOccurs="0"/>
  </sequence>
</complexType>
<complexType name="slaInfoType">
  <sequence>
    <element name="sla" type="registry:slaType"
    maxOccurs="unbounded"/>
```

```
  </sequence>
</complexType>
<complexType name="slaType">
  <simpleContent>
    <extension base="decimal">
      <attribute name="type" type="string"
      use="required"/>
      <attribute name="subtype" type="string"
      use="optional"/>
      <attribute name="command" type="string"
      use="optional"/>
      <attribute name="unit" type="string"
      use="optional"/>
    </extension>
  </simpleContent>
</complexType>
<complexType name="fieldsType">
  <sequence>
    <element name="field" type="token"
       maxOccurs="unbounded"/>
  </sequence>
  <attribute name="type" use="required">
    <simpleType>
      <restriction base="token">
        <enumeration value="shared"/>
        <enumeration value="sync"/>
      </restriction>
    </simpleType>
  </attribute>
</complexType>
<complexType name="zoneMemberType">
  <simpleContent>
    <extension base="eppcom:labelType">
      <attribute name="type" use="required">
        <simpleType>
          <restriction base="token">
            <enumeration value="primary"/>
            <enumeration value="primaryBasedOnCrDate"/>
            <enumeration value="alternate"/>
            <enumeration value="equal"/>
          </restriction>
        </simpleType>
      </attribute>
    </extension>
  </simpleContent>
</complexType>
<complexType name="relatedType">
  <sequence>
    <element name="fields" type="registry:fieldsType"
      minOccurs="0"/>
    <element name="zoneMember" type="registry:zoneMemberType"
    maxOccurs="unbounded"/>
  </sequence>
</complexType>
<complexType name="servicesType">
  <sequence>
    <element name="objURI" type="registry:uriType"
    maxOccurs="unbounded"/>
    <element name="svcExtension"
```

```
    type="registry:svcExtensionType"
    minOccurs="0"/>
  </sequence>
</complexType>
<complexType name="svcExtensionType">
  <sequence>
    <element name="extURI" type="registry:uriType"
    minOccurs="0" maxOccurs="unbounded"/>
  </sequence>
</complexType>
<complexType name="phaseType">
  <sequence>
    <element name="startDate" type="dateTime"/>
    <element name="endDate" type="dateTime"
    minOccurs="0"/>
  </sequence>
  <attribute name="type" use="required">
    <simpleType>
      <restriction base="token">
        <enumeration value="pre-delegation"/>
        <enumeration value="pre-launch"/>
        <enumeration value="sunrise"/>
        <enumeration value="landrush"/>
        <enumeration value="claims"/>
        <enumeration value="open"/>
        <enumeration value="custom"/>
      </restriction>
    </simpleType>
  </attribute>
  <attribute name="mode" default="fcfs">
    <simpleType>
      <restriction base="token">
        <enumeration value="fcfs"/>
        <enumeration value="pending-registration"/>
        <enumeration value="pending-application"/>
      </restriction>
    </simpleType>
  </attribute>
  <attribute name="name" use="optional" type="token"/>
</complexType>
<complexType name="uriType">
  <simpleContent>
    <extension base="anyURI">
      <attribute name="required" type="boolean" use="required"/>
    </extension>
  </simpleContent>
</complexType>
<complexType name="reservedNamesType">
  <choice>
    <element name="reservedName" type="normalizedString"
    minOccurs="0" maxOccurs="unbounded"/>
    <element name="reservedNameURI" type="anyURI"
    minOccurs="0"/>
  </choice>
</complexType>
<complexType name="domainNameType">
  <sequence>
    <element name="minLength" type="unsignedShort"
    minOccurs="0"/>
```

```
    <element name="maxLength" type="unsignedShort"
    minOccurs="0"/>
    <element name="alphaNumStart" type="boolean"
    minOccurs="0" default="false"/>
    <element name="alphaNumEnd" type="boolean"
    minOccurs="0" default="false"/>
    <element name="onlyDnsChars" type="boolean"
    minOccurs="0" default="true"/>
    <element name="regex" type="registry:regexType"
    minOccurs="0" maxOccurs="unbounded"/>
    <element name="reservedNames"
    type="registry:reservedNamesType"
    minOccurs="0"/>
  </sequence>
  <attribute name="level" use="required">
    <simpleType>
      <restriction base="unsignedShort">
        <minInclusive value="2"/>
      </restriction>
    </simpleType>
  </attribute>
</complexType>
<complexType name="regexType">
  <sequence>
    <element name="expression" type="string"/>
    <element name="explanation" minOccurs="0">
      <complexType>
        <simpleContent>
          <extension base="normalizedString">
            <attribute name="lang" type="language"
            default="en"/>
          </extension>
        </simpleContent>
      </complexType>
    </element>
  </sequence>
</complexType>
<simpleType name="variantStrategyType">
  <restriction base="token">
    <enumeration value="blocked"/>
    <enumeration value="restricted"/>
    <enumeration value="open"/>
  </restriction>
</simpleType>
<complexType name="languageType">
  <sequence>
    <element name="table" type="anyURI"
    minOccurs="0"/>
    <element name="variantStrategy"
    type="registry:variantStrategyType"
    minOccurs="0"/>
  </sequence>
  <attribute name="code" type="language" use="required"/>
</complexType>
<complexType name="idnType">
  <sequence>
    <element name="idnVersion" type="token"
    minOccurs="0"/>
    <element name="idnaVersion" type="token"/>
```

```
    <element name="unicodeVersion" type="token"/>
    <element name="encoding" type="token"
    minOccurs="0" default="Punycode"/>
    <element name="commingleAllowed" type="boolean"
    minOccurs="0" default="false"/>
    <element name="language" type="registry:languageType"
    minOccurs="0" maxOccurs="unbounded"/>
  </sequence>
</complexType>
<complexType name="dContactType">
  <complexContent>
    <extension base="registry:minMaxType">
      <attribute name="type" use="required">
        <simpleType>
          <restriction base="token">
            <enumeration value="admin"/>
            <enumeration value="billing"/>
            <enumeration value="tech"/>
          </restriction>
        </simpleType>
      </attribute>
    </extension>
  </complexContent>
</complexType>
<complexType name="minMaxType">
  <sequence>
    <element name="min" type="unsignedShort"/>
    <element name="max" type="unsignedShort"
    minOccurs="0"/>
  </sequence>
</complexType>
<complexType name="minMaxPeriod">
  <sequence>
    <element name="min" type="registry:periodType"/>
    <element name="max" type="registry:periodType"/>
    <element name="default" type="registry:periodType"/>
  </sequence>
</complexType>
<complexType name="dPeriodType">
  <choice>
    <element name="length" type="registry:minMaxPeriod"/>
    <element name="serverDecided">
      <complexType/>
    </element>
  </choice>
  <attribute name="command" type="token"
  use="required"/>
</complexType>
<complexType name="gPeriodType">
  <simpleContent>
    <extension base="registry:periodType">
      <attribute name="command" type="token"
      use="required"/>
    </extension>
  </simpleContent>
</complexType>
<complexType name="periodType">
  <simpleContent>
    <extension base="unsignedShort">
```

```
      <attribute name="unit" type="registry:pUnitType"
      use="required"/>
    </extension>
  </simpleContent>
</complexType>
<simpleType name="pUnitType">
  <restriction base="token">
    <enumeration value="y"/>
    <enumeration value="m"/>
    <enumeration value="d"/>
    <enumeration value="h"/>
  </restriction>
</simpleType>
<complexType name="rgpType">
  <sequence>
    <element name="redemptionPeriod"
    type="registry:periodType"/>
    <element name="pendingRestore"
    type="registry:periodType"/>
    <element name="pendingDelete"
    type="registry:periodType"/>
  </sequence>
</complexType>
<complexType name="keyInterfaceType">
  <sequence>
    <element name="min" type="unsignedShort"/>
    <element name="max" type="unsignedShort"/>
    <element name="alg" type="token" minOccurs="0"
    maxOccurs="unbounded"/>
  </sequence>
</complexType>
<complexType name="dsInterfaceType">
  <complexContent>
    <extension base="registry:keyInterfaceType">
      <sequence>
        <element name="digestType" type="token"
        minOccurs="0"
        maxOccurs="unbounded"/>
      </sequence>
    </extension>
  </complexContent>
</complexType>
<complexType name="maxSigLifeType">
  <sequence>
    <element name="clientDefined" type="boolean"
    minOccurs="0" default="false"/>
    <element name="default" type="int"
    minOccurs="0"/>
    <element name="min" type="int"
    minOccurs="0"/>
    <element name="max" type="int"
    minOccurs="0"/>
  </sequence>
</complexType>
<complexType name="dnssecType">
  <sequence>
    <choice>
      <element name="dsDataInterface"
      type="registry:dsInterfaceType"/>
```

```
      <element name="keyDataInterface"
      type="registry:keyInterfaceType"/>
    </choice>
    <element name="maxSigLife"
    type="registry:maxSigLifeType"/>
    <element name="urgent" type="boolean"
    minOccurs="0" default="false"/>
  </sequence>
</complexType>
<complexType name="supportedStatusType">
  <sequence>
    <element name="status" type="token" minOccurs="1"
    maxOccurs="unbounded"/>
  </sequence>
</complexType>

<complexType name="keyValuesType">
  <sequence>
    <element name="value" maxOccurs="unbounded">
      <complexType>
        <simpleContent>
          <extension base="normalizedString">
            <attribute name="key" type="token"
            use="required"/>
          </extension>
        </simpleContent>
      </complexType>
    </element>
  </sequence>
</complexType>
<complexType name="domainType">
  <sequence>
    <element name="domainName" type="registry:domainNameType"
    maxOccurs="unbounded"/>
    <element name="idn" type="registry:idnType"
    minOccurs="0"/>
    <element name="premiumSupport" type="boolean"
    minOccurs="0" default="false"/>
    <element name="contactsSupported" type="boolean"
    minOccurs="0" default="true"/>
    <element name="contact" type="registry:dContactType"
    minOccurs="0" maxOccurs="3"/>
    <element name="ns" type="registry:minMaxType"/>
    <element name="childHost" type="registry:minMaxType"/>
    <element name="period" type="registry:dPeriodType"
    minOccurs="0" maxOccurs="unbounded"/>
    <element name="transferHoldPeriod" type="registry:periodType"/>
    <element name="gracePeriod" type="registry:gPeriodType"
    minOccurs="0" maxOccurs="unbounded"/>
    <element name="rgp" type="registry:rgpType"
    minOccurs="0"/>
    <element name="dnssec" type="registry:dnssecType"
    minOccurs="0"/>
    <element name="maxCheckDomain" type="unsignedShort"/>
    <element name="supportedStatus" type="registry:supportedStatusType"
    minOccurs="0"/>
    <element name="authInfoRegex" type="registry:regexType"
    minOccurs="0"/>
    <element name="customData" type="registry:keyValuesType"
```

```
    minOccurs="0"/>
  </sequence>
</complexType>
<simpleType name="intHostSharePolicyType">
  <restriction base="token">
    <enumeration value="perZone"/>
    <enumeration value="perSystem"/>
  </restriction>
</simpleType>
<simpleType name="extHostSharePolicyType">
  <restriction base="token">
    <enumeration value="perRegistrar"/>
    <enumeration value="perZone"/>
    <enumeration value="perSystem"/>
  </restriction>
</simpleType>
<complexType name="intHostPolicyType">
  <sequence>
    <element name="minIP" type="unsignedShort"/>
    <element name="maxIP" type="unsignedShort"/>
    <element name="sharePolicy"
       type="registry:intHostSharePolicyType" minOccurs="0"/>
  </sequence>
</complexType>
<complexType name="extHostPolicyType">
  <sequence>
    <element name="minIP" type="unsignedShort"/>
    <element name="maxIP" type="unsignedShort"/>
    <element name="sharePolicy"
       type="registry:extHostSharePolicyType" minOccurs="0"/>
  </sequence>
</complexType>
<complexType name="hostType">
  <sequence>
    <element name="internal" type="registry:intHostPolicyType"/>
    <element name="external" type="registry:extHostPolicyType"/>
    <element name="nameRegex" type="registry:regexType"
    minOccurs="0" maxOccurs="unbounded"/>
    <element name="maxCheckHost" type="unsignedShort"/>
    <element name="supportedStatus" type="registry:supportedStatusType"
    minOccurs="0"/>
    <element name="customData" type="registry:keyValuesType"
    minOccurs="0"/>
  </sequence>
</complexType>
<complexType name="minMaxLength">
  <sequence>
    <element name="minLength" type="unsignedShort"/>
    <element name="maxLength" type="unsignedShort"/>
  </sequence>
</complexType>
<simpleType name="contactSharePolicyType">
  <restriction base="token">
    <enumeration value="perZone"/>
    <enumeration value="perSystem"/>
  </restriction>
</simpleType>
<complexType name="streetType">
  <complexContent>
```

```
      <extension base="registry:minMaxLength">
        <sequence>
          <element name="minEntry" type="unsignedShort"/>
          <element name="maxEntry" type="unsignedShort"/>
        </sequence>
      </extension>
    </complexContent>
  </complexType>
  <complexType name="contactAddressType">
    <sequence>
      <element name="street" type="registry:streetType"/>
      <element name="city" type="registry:minMaxLength"/>
      <element name="sp" type="registry:minMaxLength"/>
      <element name="pc" type="registry:minMaxLength"/>
    </sequence>
  </complexType>
  <complexType name="postalType">
    <sequence>
      <element name="name" type="registry:minMaxLength"/>
      <element name="org" type="registry:minMaxLength"/>
      <element name="address" type="registry:contactAddressType"/>
      <element name="voiceRequired" type="boolean"
      minOccurs="0" default="false"/>
      <element name="voiceExt" type="registry:minMaxLength"
      minOccurs="0"/>
      <element name="faxExt" type="registry:minMaxLength"
      minOccurs="0"/>
      <element name="emailRegex" type="registry:regexType"
      minOccurs="0" maxOccurs="unbounded"/>
    </sequence>
  </complexType>
  <complexType name="contactType">
    <sequence>
      <element name="contactIdRegex" type="registry:regexType"
          minOccurs="0"/>
      <element name="sharePolicy"
          type="registry:contactSharePolicyType" minOccurs="0"/>
      <element name="intSupport" type="boolean"/>
      <element name="locSupport" type="boolean"/>
      <element name="postalInfo" type="registry:postalType"/>
      <element name="maxCheckContact" type="unsignedShort"/>
      <element name="authInfoRegex" type="registry:regexType"
      minOccurs="0"/>
      <element name="clientDisclosureSupported" type="boolean"
      minOccurs="0" default="false"/>
      <element name="supportedStatus" type="registry:supportedStatusType"
      minOccurs="0"/>
      <element name="transferHoldPeriod" type="registry:periodType"
      minOccurs="0"/>
      <element name="customData" type="registry:keyValuesType"
      minOccurs="0"/>
    </sequence>
  </complexType>
</schema>
```
END

5 References

Document all references.

[RFC5730] Hollenbeck, S., "Extensible Provisioning Protocol (EPP)", RFC 5730, August 2009.

[RFC5731] Hollenbeck, S., " Extensible Provisioning Protocol Domain Name Mapping", RFC 5731, August 2009.

[RFC5732] Hollenbeck, S., " Extensible Provisioning Protocol Host Mapping", RFC 5732, August 2009.

[RFC5733] Hollenbeck, S., " Extensible Provisioning Protocol Contact Mapping", RFC 5733, August 2009.

[W3C.REC-xmlschema-2-20041028] Maloney, M., Thompson, H., Mendelsohn, N., and D. Beech, "XML Schema Part 1: Structures Second Edition", World Wide Web Consortium Recommendation REC-xmlschema-1-20041028, October 2004.

[RFC0952] K. Harrenstien et al.: "DOD Internet Host Table Specification", RFC 952, October 1985.

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

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

[X3C.REC-xml-20040204] Sperberg-McQueen, C., Maler, E., Yergeau, F., Paoli, J., and T. Bray, "Extensible Markup Language (XML) 1.0 (Third Edition)", World Wide Web Consortium FirstEdition REC-xml-20040204, February 2004.

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

[W3C.REC-xmlschema-2-20041028] Malhotra, A. and P. Biron, "XML Schema Part 2: Datatypes Second Edition", World Wide Web Consortium Recommendation REC-xmlschema-2-20041028, October 2004.