Network Working Group
Internet-Draft
Intended status: Standards Track
Expires: February 18, 2019

J. Gould
VeriSign, Inc.

August 17, 2018

Launch Phase Policy Extensions Mapping for the Extensible Provisioning Protocol (EPP) draft-gould-regext-launch-policy-01

Abstract

This document describes an Extensible Provisioning Protocol (EPP) extension of the Registry Mapping to define the server policy of the Launch Phase EPP extension. The server policy of the Launch Phase EPP extension includes the MAYs, SHOULDs, and options implemented by the server.

Status of This Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at http://datatracker.ietf.org/drafts/current/.

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

This Internet-Draft will expire on February 18, 2019.

Table of Contents

1. Introduction 2
1.1. Conventions Used in This Document 3
2. Object Attributes 3
2.1. Dates and Times 3
2.2. Zone Object 3
3. EPP Command Mapping 12
3.1. EPP Query Commands 12
3.1.1. EPP <check> Command 12
3.1.2. EPP <info> Command 12
3.1.3. EPP <transfer> Query Command 13
3.2. EPP Transform Commands 13
3.2.1. EPP <create> Command 14
3.2.2. EPP <delete> Command 15
3.2.3. EPP <renew> Command 15
3.2.4. EPP <transfer> Command 15
3.2.5. EPP <update> Command 15
4. Formal Syntax 16
4.1. Launch Policy Schema 16
5. IANA Considerations 20
5.1. XML Namespace 20
5.2. EPP Extension Registry 20
6. Implementation Status 21
7. Security Considerations 21
8. Acknowledgements 21
9. References 21
9.1. Normative References 21
9.2. Informative References 22
9.3. URIs 22
Appendix A. Change History 22
A.1. Change from 00 to 01 22
Author's Address 22

1. Introduction

This document describes an extension mapping for version 1.0 of the Extensible Provisioning Protocol (EPP) [RFC5730]. This document describes an extension of the Registry Mapping to define the server policy of the Launch Phase EPP extension [RFC8334] for a registry zone (e.g., top-level domain). A registry zone, also referred to as a "zone" in this document, is a domain name that the Domain Name Registry supports provisioning operations to manage. The extension enables provisioning of the registry zone policy in the Domain Name

Registry. A Domain Name Registry MAY support a subset of all of the commands extended in this extension.

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 RFC 2119 [RFC2119].

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

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.

The XML namespace prefix "lp" is used for the namespace "urn:ietf:params:xml:ns:epp:launchPolicy-0.1", but implementations MUST NOT depend on it and instead employ a proper namespace-aware XML parser and serializer to interpret and output the XML documents.

2. Object Attributes

An EPP launch phase policy 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. 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 XML Schema Part 2 [1] 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.

2.2. Zone Object

The Zone object, represented by the <registry:zone> element in the Registry Mapping, is the object that is extended by this extension with the <lp:zone> element. The Zone object can apply to any zone level (top level, second level, third level, etc.). The <lp:zone>

element contains zero or more <lp:phase> elements, ordered by ascending start date.

The <lp:phase> element includes the following attributes:

"type": Attribute that defines the phase name / type that include the following possible values:

"pre-delegation" Phase when pre-delegation testing is done.
"pre-launch" 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" Custom launch phase that is defined using the "name" attribute.

"name": OPTIONAL identifier attribute, represented in the 7-bit US-ASCII character set, that defines the phase name when the "type" attribute is set to "custom" or the sub-phase name when the "type" attribute is set to a non-"custom" value.

"mode": OPTIONAL attribute that defines how the phase operates with the following possible values and with the default value of "fcfs":

"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 the "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 <lp:phase> element contains the following child elements:

<lp:startDate>: The Start date and time for the phase.
<lp:endDate>: The OPTIONAL end date and time for the phase.
<lp:validatePhase>: An OPTIONAL boolean value that indicates whether the server validates the phase provided by the client in the <launch:phase> element.
<lp:validatorId>: Zero or more <lp:validatorId> elements that define the supported Validator Identifier values for the phase. An example is the reserved Validator Identifier of "tmch".
<lp:status>: Zero or more <lp:status> elements that defines the supported launch status values, in precedence order, for the phase. The <lp:status> element has the required "s" attribute with one of the possible the status values. The OPTIONAL "name" attribute is an identifier, represented in the 7-bit US-ASCII character set, that is used to define the name of the "custom" status. When the "s" attribute is set to "custom", then the "name" attribute MUST be set. The <lp:status> element text MAY provide the status value description, and the OPTIONAL "lang" attribute MAY be present to identify the language of the description if the negotiated value is something other than the default value of "en" (English). The possible values of the "s" attribute include:

"pendingValidation" The initial state of a newly-created application or registration object.
"validated" The application or registration meets relevant registry rules.
"invalid" The application or registration does not validate according to registry rules.
"pendingAllocation" The allocation of the application or registration is pending based on the results of some out-of-band process (for example, an auction).
"allocated" The object corresponding to the application or registration has been provisioned.
"rejected" The application or registration object was not provisioned.
"custom" A custom status that is defined using the "name" attribute.
<lp:pendingCreate>: An OPTIONAL boolean value that indicates that the server will place the Launch Application or the Launch

Registration in the "pendingCreate" status as specified in [RFC5731].

<lp:pollPolicy>: An OPTIONAL element that defines the poll messaging policy for the phase. The <lp:pollPolicy> element contains the following child elements:

<lp:intermediateStatus>: A boolean value indicating whether the server will insert poll messages, per [RFC5730], for the applicable intermediate statuses, including the "pendingValidation", "validated", "pendingAllocation", and "invalid" statuses, using the <domain:infData> element with the <launch:infData> extension.

<lp:nonMandatoryInfo>: A boolean value indicating whether the server will include non-mandatory information in the <domain:infData> element of the poll message.

<lp:extensionInfo>: A boolean value indicating whether the server will include further extensions that would normally be included in the response to the <domain:info> command, per [RFC5731], in the poll message.

<lp:markValidation>: Zero to four <lp:markValidation> elements that defines the supported Mark Validation Models supported by the phase. The <lp:markValidation> element has the following possible values:

"code" Indicates support for the "code" Mark Validation Model, where the mark code by itself is used to validate that the mark matches the domain name. This model is supported using the <launch:codeMark> element with just the <launch:code> element.

"mark" Indicates support for the "mark" Mark Validation Model, where the mark information is passed without any other validation element. The server will use some custom form of validation to validate that the mark information is authentic. This model is supported using the <launch:codeMark> element with just the <mark:mark> element.

"codeWithMark" Indicates support for the "code with mark" Mark Validation Model, where the code is used along with the mark information by the server to validate the mark utilizing an external party. This model is supported using the <launch:codeMark> element that contains both the <launch:code> and the <mark:mark> elements.

"signedMark" Indicates support for the "signed mark" Mark Validation Model, where the mark information is digitally signed. The digital signature can be directly validated by the server using the public key of the external party

that created the signed mark using its private key. This model is supported using the <smd:signedMark> and <smd:encodedSignedMark> elements.

<lp:maxMarks>: An OPTIONAL maximum number of marks per domain name for the phase.

<lp:markSupported>: Zero or more <lp:markSupported> elements that defines the XML namespace of the marks that are supported in the phase. For example, the XML namespace "urn:ietf:params:xml:ns:mark-1.0" for [RFC7848].

<lp:signedMarkSupported>: Zero or more <lp:signedMarkSupported> elements that defines the XML namespace of the signed marks that are supported in the phase. For example, the XML namespace "urn:ietf:params:xml:ns:signedMark-1.0" for [RFC7848].

<lp:encodedSignedMarkSupported>: Zero or more <lp:encodedSignedMarkSupported> elements that defines the XML namespace of the encoded signed marks that are supported in the phase. For example, the XML namespace "urn:ietf:params:xml:ns:signedMark-1.0" for [RFC7848].

<lp:checkForm>: Zero to three <lp:checkForm> elements that defines the supported check forms. The <lp:checkForm> element has the following possible values:

"claims" Indicates support for the Claims Check Form, which defines a new command called the Claims Check Command that is used to determine whether or not there are any matching trademarks, in the specified launch phase, for each domain name passed in the command.

"availability" Indicates support for the Availability Check Form, which extends the Domain Check Command to specify which launch phase to use to check the availability for each domain name passed in the command.

"trademark" Indicates support for the Trademark Check Form, which defines a new command called the Trademark Check Command that is used to determine whether or not there are any matching trademarks for each domain name passed in the command, independent of the active launch phase.

<lp:infoPhase>: Zero or more <lp:infoPhase> elements that defines the possible <launch:phase> values that can be passed by the client in the phase. The <lp:infoPhase> supports the "type" and "name" attributes defined for the <lp:phase> element.

<lp:createForm>: Zero to three <lp:createForm> elements that defines the supported create forms. The <lp:createForm> element has the following possible values:

"sunrise" Indicates support for the Sunrise Create Form, which is an extension of the Domain Create Command to include the verifiable trademark information that the server uses to match against the domain name to authorize the domain create.
"claims" Indicates support for the Claims Create Form, which is an extension of the Domain Create Command to include information related to the registrant's acceptance of the Claims Notice.
"general" Indicates support for the General Create Form, which is an extension of the Domain Create Command to specify the target launch phase for the domain create.
"mixed" Indicates support for the Mixed Create Form, where a mix of create forms is supported. For example, the Sunrise Create Form and the Claims Create Form is supported in a single command by including both the verified trademark information and the information related to the registrant's acceptance of the Claims Notice.

<lp:createValidateType>: An OPTIONAL boolean value that indicates whether the server validates the OPTIONAL <launch:create> "type" attribute.

Example of a <lp:zone> element with six launch phases, including "sunrise", "claims"/"lrp1", "claims"/"landrush", "claims"/"open", "lrp2", and "open":

```
<lp:zone>
  <lp:phase
    type="sunrise"
    mode="pending-application"
  >
    <lp:startDate>2017-11-01T00:00:00.0Z
    </lp:startDate>
    <lp:endDate>2017-12-01T00:00:00.0Z
    </lp:endDate>
    <lp:validatePhase>true</lp:validatePhase>
    <lp:validatorId>tmch
    </lp:validatorId>
    <lp:status s="pendingAllocation"/>
    <lp:status s="allocated"/>
    <lp:status s="rejected"/>
    <lp:pollPolicy>
      <lp:intermediateStatus>false</lp:intermediateStatus>
      <lp:nonMandatoryInfo>false</lp:nonMandatoryInfo>
      <lp:extensionInfo>false</lp:extensionInfo>
    </lp:pollPolicy>
    <lp:markValidation>signedMark</lp:markValidation>
    <lp:maxMarks>1</lp:maxMarks>
```

```
      <lp:signedMarkSupported>
        urn:ietf:params:xml:ns:signedMark-1.0
      </lp:signedMarkSupported>
      <lp:encodedSignedMarkSupported>
        urn:ietf:params:xml:ns:signedMark-1.0
      </lp:encodedSignedMarkSupported>
      <lp:infoPhase type="sunrise"/>
      <lp:createForm>sunrise</lp:createForm>
      <lp:createValidateType>true
      </lp:createValidateType>
    </lp:phase>
    <lp:phase
      type="claims"
      name="lrp1"
      mode="pending-registration"
    >
      <lp:startDate>2017-12-01T00:00:00.0Z
      </lp:startDate>
      <lp:endDate>2017-12-08T00:00:00.0Z
      </lp:endDate>
      <lp:validatePhase>true</lp:validatePhase>
      <lp:validatorId>tmch
      </lp:validatorId>
      <lp:status s="pendingValidation"/>
      <lp:status s="allocated"/>
      <lp:status s="rejected"/>
      <lp:pendingCreate>true</lp:pendingCreate>
      <lp:checkForm>claims</lp:checkForm>
      <lp:checkForm>availability</lp:checkForm>
      <lp:checkForm>trademark</lp:checkForm>
      <lp:infoPhase type="sunrise"/>
      <lp:infoPhase
        type="claims"
        name="lrp1"/>
      <lp:createForm>claims</lp:createForm>
      <lp:createValidateType>true
      </lp:createValidateType>
    </lp:phase>
    <lp:phase
      type="claims"
      name="landrush"
      mode="pending-application"
    >
      <lp:startDate>2017-12-08T00:00:00.0Z
      </lp:startDate>
      <lp:endDate>2017-12-15T00:00:00.0Z
      </lp:endDate>
      <lp:validatePhase>true</lp:validatePhase>
```

```
      <lp:validatorId>tmch
      </lp:validatorId>
      <lp:status s="pendingAllocation"/>
      <lp:status s="allocated"/>
      <lp:status s="rejected"/>
      <lp:pendingCreate>true</lp:pendingCreate>
      <lp:pollPolicy>
        <lp:intermediateStatus>false</lp:intermediateStatus>
        <lp:nonMandatoryInfo>false</lp:nonMandatoryInfo>
        <lp:extensionInfo>false</lp:extensionInfo>
      </lp:pollPolicy>
      <lp:checkForm>claims</lp:checkForm>
      <lp:checkForm>availability</lp:checkForm>
      <lp:checkForm>trademark</lp:checkForm>
      <lp:infoPhase
        type="claims"
        name="lrp1"/>
      <lp:infoPhase
        type="claims"
        name="landrush"/>
      <lp:createForm>claims</lp:createForm>
      <lp:createValidateType>true
      </lp:createValidateType>
    </lp:phase>
    <lp:phase
      type="claims"
      name="open"
      mode="fcfs"
    >
      <lp:startDate>2017-12-15T00:00:00.0Z
      </lp:startDate>
      <lp:endDate>2018-02-15T00:00:00.0Z
      </lp:endDate>
      <lp:validatePhase>true</lp:validatePhase>
      <lp:validatorId>tmch
      </lp:validatorId>
      <lp:checkForm>claims</lp:checkForm>
      <lp:checkForm>availability</lp:checkForm>
      <lp:checkForm>trademark</lp:checkForm>
      <lp:infoPhase
        type="claims"
        name="landrush"/>
      <lp:infoPhase
        type="claims"
        name="open"/>
      <lp:createForm>claims</lp:createForm>
      <lp:createForm>general</lp:createForm>
      <lp:createValidateType>true
```

```
      </lp:createValidateType>
    </lp:phase>
    <lp:phase
      type="custom"
      name="lrp2"
      mode="pending-registration"
    >
      <lp:startDate>2018-02-15T00:00:00.0Z
      </lp:startDate>
      <lp:endDate>2018-03-15T00:00:00.0Z
      </lp:endDate>
      <lp:validatePhase>true</lp:validatePhase>
      <lp:validatorId>lrp2-custom
      </lp:validatorId>
      <lp:status
        s="custom"
        name="pendingInternalValidation"
      >
        Internally validate registration
      </lp:status>
      <lp:status
        s="custom"
        name="pendingExternalValidation"
      >
        Externally validate registration
      </lp:status>
      <lp:status s="allocated"/>
      <lp:status s="rejected"/>
      <lp:pendingCreate>true</lp:pendingCreate>
      <lp:pollPolicy>
        <lp:intermediateStatus>true</lp:intermediateStatus>
        <lp:nonMandatoryInfo>false</lp:nonMandatoryInfo>
        <lp:extensionInfo>false</lp:extensionInfo>
      </lp:pollPolicy>
      <lp:markValidation>signedMark</lp:markValidation>
      <lp:maxMarks>1</lp:maxMarks>
      <lp:signedMarkSupported>
        http://www.example.com/epp/lrp2-custom-1.0
      </lp:signedMarkSupported>
      <lp:encodedSignedMarkSupported>
        http://www.example.com/epp/lrp2-custom-1.0
      </lp:encodedSignedMarkSupported>
      <lp:infoPhase
        type="claims"
        name="open"/>
      <lp:infoPhase
        type="custom"
        name="lrp2"/>
```

```
      <lp:createForm>sunrise</lp:createForm>
      <lp:createForm>general</lp:createForm>
      <lp:createForm>claims</lp:createForm>
      <lp:createForm>mixed</lp:createForm>
      <lp:createValidateType>true
      </lp:createValidateType>
    </lp:phase>
    <!-- Use default mode="fcfs" -->
    <lp:phase type="open">
      <lp:startDate>2018-03-15T00:00:00.0Z
      </lp:startDate>
      <lp:validatePhase>false</lp:validatePhase>
    </lp:phase>
  </lp:zone>
```

3. EPP Command Mapping

A detailed description of the EPP syntax and semantics can be found in the EPP core protocol specification [RFC5730]. The command mappings described here are specifically for use in provisioning and managing launch phase policy via EPP.

3.1. EPP Query Commands

EPP [RFC5730] provides three commands to retrieve object information: <check> to determine if an object is known to the server, <info> to retrieve detailed information associated with an object, and <transfer> to retrieve object transfer status information.

3.1.1. EPP <check> Command

This extension does not define any extension of the EPP <check> command or response described in the Registry Mapping.

3.1.2. EPP <info> Command

This extension does not add any elements to the EPP <info> command described in the Registry Mapping. However, additional elements are defined for the <info> response to a query by the fully qualified name of the zone object.

When an <info> command has been processed successfully, the EPP <resData> element MUST contain a child elements as described in the Registry Mapping. In addition, the EPP <extension> element SHOULD contain a child <lp:infData> element that identifies the extension namespace if the zone object has data associated with this extension and based on server policy. The <lp:infData> element contains the following child elements:

<lp:zone>: Element that contains the full set of launch phase policy attributes for the zone as defined in Section 2.2.

Example <info> response to query for the full set of "EXAMPLE" zone object attributes including the launch phase policy attributes:

```
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:infData
S:        xmlns:registry="urn:ietf:params:xml:ns:registry-0.1">
S:      ...
S:      </registry:infData>
S:    </resData>
S:    <extension>
S:      <lp:infData
S:        xmlns:lp="urn:ietf:params:xml:ns:epp:launchPolicy-0.1">
S:        ...
S:      </lp:infData>
S:    </extension>
S:    <trID>
S:      <clTRID>ABC-12345</clTRID>
S:      <svTRID>54322-XYZ</svTRID>
S:    </trID>
S:  </response>
S:</epp>
```

3.1.3. EPP <transfer> Query Command

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

3.2. EPP Transform Commands

EPP provides five commands to transform objects: <create> to create an instance of an object, <delete> to delete an instance of an object, <renew> to extend the validity period of an object, <transfer> to manage object sponsorship changes, and <update> to change information associated with an object.

3.2.1. EPP <create> Command

This extension defines additional elements for the EPP <create> command described in the Registry Mapping. No additional elements are defined for the EPP <create> response.

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 described in the Registry Mapping, the command MUST contain an <extension> element, and the <extension> element MUST contain a <lp:create> element that identifies the extension namespace if the client wants to associate data in this extension to the zone object. The <lp:create> element contains the following child elements:

<lp:zone>: Element that contains the full set of launch phase policy attributes for the zone to create as defined in Section 2.2.

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="urn:ietf:params:xml:ns:epp:registry-0.1">
C:        <registry:zone>
C:          <registry:name>EXAMPLE</registry:name>
C:          ...
C:        </registry:zone>
C:      </registry:create>
C:    </create>
C:    <extension>
C:      <lp:create
C:        xmlns:lp="urn:ietf:params:xml:ns:epp:launchPolicy-0.1">
C:        ...
C:      </lp:create>
C:    </extension>
C:    <clTRID>ABC-12345</clTRID>
C:  </command>
C:</epp>
```

When a <create> command has been processed successfully, the EPP response is as described in the Registry Mapping.

3.2.2. EPP <delete> Command

This extension does not add any elements to the EPP <delete> command or <delete> response described in the Registry Mapping.

3.2.3. EPP <renew> Command

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

3.2.5. EPP <update> Command

This extension defines additional elements for the EPP <update> command described in the Registry Mapping. No additional elements are defined for the EPP <update> response.

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 described in the Registry Mapping, the command MUST contain an <extension> element, and the <extension> element MUST contain a <lp:update> element that identifies the extension namespace if the client wants to associate data in this extension to the zone object. The <lp:update> element contains the following child elements:

<lp:zone>: One or more elements that contain the full set of attributes for the zones launch phase policy as defined in Section 2.2. The update completely replaces the prior version of the zone launch phase policy.

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:      <registry:update
C:        xmlns:registry="urn:ietf:params:xml:ns:epp:registry-0.1">
C:        <registry:zone>
C:          <registry:name>EXAMPLE</registry:name>
C:          ...
C:        </registry:zone>
C:      </registry:update>
C:    </update>
C:    <extension>
C:      <lp:update
C:        xmlns:lp="urn:ietf:params:xml:ns:epp:launchPolicy-0.1">
C:        ...
C:      </lp:update>
C:    </extension>
C:    <clTRID>ABC-12345</clTRID>
C:  </command>
C:</epp>
```

When an extended <update> command has been processed successfully, the EPP response is as described in the Registry Mapping.

4. Formal Syntax

One schema presented here is the EPP Launch Phase Policy Schema.

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.

4.1. Launch Policy Schema

```
BEGIN
<?xml version="1.0" encoding="UTF-8"?>
<schema xmlns="http://www.w3.org/2001/XMLSchema"
  xmlns:launchPolicy="urn:ietf:params:xml:ns:epp:launchPolicy-0.1"
  targetNamespace="urn:ietf:params:xml:ns:epp:launchPolicy-0.1"
  elementFormDefault="qualified">
  <annotation>
    <documentation>Extensible Provisioning Protocol v1.0
    Launch Phase Policy Extension Schema.</documentation>
```

```
    </annotation>
    <!--
      Child elements found in EPP commands.
    -->
    <element name="create" type="launchPolicy:zoneContainerType" />
    <element name="update" type="launchPolicy:zoneContainerType" />
    <!--
      Child response elements.
    -->
    <element name="infData" type="launchPolicy:zoneContainerType" />
    <!--
      Container for zone launch phase policy
    -->
    <complexType name="zoneContainerType">
      <sequence>
        <element name="zone" type="launchPolicy:zoneType" />
      </sequence>
    </complexType>
    <!--
      Zone launch phase policies
    -->
    <complexType name="zoneType">
      <sequence>
        <element name="phase" type="launchPolicy:phaseType"
          minOccurs="0" maxOccurs="unbounded" />
      </sequence>
    </complexType>
    <complexType name="phaseNameType">
      <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="name" use="optional" type="token" />
    </complexType>
    <!--
      <phase> elements.
    -->
    <complexType name="phaseType">
      <complexContent>
```

```
        <extension base="launchPolicy:phaseNameType">
          <sequence>
            <element name="startDate" type="dateTime" />
            <element name="endDate" type="dateTime"
              minOccurs="0" />
            <element name="validatePhase" type="boolean"
              minOccurs="0" />
            <element name="validatorId" type="token"
              minOccurs="0" maxOccurs="unbounded" />
            <element name="status" type="launchPolicy:statusType"
              minOccurs="0" maxOccurs="unbounded" />
            <element name="pendingCreate" type="boolean"
              minOccurs="0" />
            <element name="pollPolicy"
              type="launchPolicy:pollPolicyType" minOccurs="0" />
            <element name="markValidation"
              type="launchPolicy:markValidationType"
              minOccurs="0" maxOccurs="4" />
            <element name="maxMarks" type="short"
              minOccurs="0" />
            <element name="markSupported" type="token"
              minOccurs="0" maxOccurs="unbounded" />
            <element name="signedMarkSupported" type="token"
              minOccurs="0" maxOccurs="unbounded" />
            <element name="encodedSignedMarkSupported" type="token"
              minOccurs="0" maxOccurs="unbounded" />
            <element name="checkForm"
              type="launchPolicy:checkFormType"
              minOccurs="0" maxOccurs="3" />
            <element name="infoPhase"
              type="launchPolicy:phaseNameType"
              minOccurs="0" maxOccurs="unbounded" />
            <element name="createForm"
              type="launchPolicy:createFormType"
              minOccurs="0" maxOccurs="4" />
            <element name="createValidateType" type="boolean"
              minOccurs="0" />
          </sequence>
          <attribute name="mode" default="fcfs">
            <simpleType>
              <restriction base="token">
                <enumeration value="fcfs" />
                <enumeration value="pending-registration" />
                <enumeration value="pending-application" />
              </restriction>
            </simpleType>
          </attribute>
        </extension>
```

```
      </complexContent>
    </complexType>
    <simpleType name="statusValueType">
      <restriction base="token">
        <enumeration value="pendingValidation" />
        <enumeration value="validated" />
        <enumeration value="invalid" />
        <enumeration value="pendingAllocation" />
        <enumeration value="allocated" />
        <enumeration value="rejected" />
        <enumeration value="custom" />
      </restriction>
    </simpleType>
    <!-- Status type definition -->
    <complexType name="statusType">
      <simpleContent>
        <extension base="normalizedString">
          <attribute name="s" type="launchPolicy:statusValueType"
            use="required" />
          <attribute name="lang" type="language" default="en" />
          <attribute name="name" type="token" />
        </extension>
      </simpleContent>
    </complexType>
    <complexType name="pollPolicyType">
      <sequence>
        <element name="intermediateStatus" type="boolean" />
        <element name="nonMandatoryInfo" type="boolean" />
        <element name="extensionInfo" type="boolean" />
      </sequence>
    </complexType>
    <simpleType name="markValidationType">
      <restriction base="token">
        <enumeration value="code" />
        <enumeration value="mark" />
        <enumeration value="codeWithMark" />
        <enumeration value="signedMark" />
      </restriction>
    </simpleType>
    <simpleType name="checkFormType">
      <restriction base="token">
        <enumeration value="claims" />
        <enumeration value="availability" />
        <enumeration value="trademark" />
      </restriction>
    </simpleType>
    <simpleType name="createFormType">
      <restriction base="token">
```

```
        <enumeration value="sunrise" />
        <enumeration value="claims" />
        <enumeration value="general" />
        <enumeration value="mixed" />
      </restriction>
    </simpleType>
  </schema>
  END
```

5. IANA Considerations

5.1. XML Namespace

This document uses URNs to describe XML namespaces and XML schemas conforming to a registry mechanism described in [RFC3688].

Registration request for the launch phase policy namespace:

URI: urn:ietf:params:xml:ns:epp:launchPolicy-0.1
Registrant Contact: IESG
XML: None. Namespace URIs do not represent an XML specification.

Registration request for the launch phase policy XML schema:

URI: urn:ietf:params:xml:schema:epp:launchPolicy-0.1
Registrant Contact: IESG
XML: See the "Formal Syntax" section of this document.

5.2. EPP Extension Registry

The EPP extension described in this document should be registered by the IANA in the EPP Extension Registry described in [RFC7451]. The details of the registration are as follows:

Name of Extension: "Launch Phase Policy Extensions Mapping for the Extensible Provisioning Protocol (EPP)"

Document status: Standards Track

Reference: (insert reference to RFC version of this document)

Registrant Name and Email Address: IESG, <iesg@ietf.org>

TLDs: Any

IPR Disclosure: None

Status: Active

Notes: None

6. Implementation Status

TBD

7. Security Considerations

The mapping extensions described in this document do not provide any security services beyond those described by EPP [RFC5730] and protocol layers used by EPP. The security considerations described in these other specifications apply to this specification as well.

8. Acknowledgements

TBD

9. References

9.1. Normative References

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <https://www.rfc-editor.org/info/rfc2119>.

[RFC3688] Mealling, M., "The IETF XML Registry", BCP 81, RFC 3688, DOI 10.17487/RFC3688, January 2004, <https://www.rfc-editor.org/info/rfc3688>.

[RFC5730] Hollenbeck, S., "Extensible Provisioning Protocol (EPP)", STD 69, RFC 5730, DOI 10.17487/RFC5730, August 2009, <https://www.rfc-editor.org/info/rfc5730>.

[RFC5731] Hollenbeck, S., "Extensible Provisioning Protocol (EPP) Domain Name Mapping", STD 69, RFC 5731, DOI 10.17487/RFC5731, August 2009, <https://www.rfc-editor.org/info/rfc5731>.

[RFC7848] Lozano, G., "Mark and Signed Mark Objects Mapping", RFC 7848, DOI 10.17487/RFC7848, June 2016, <https://www.rfc-editor.org/info/rfc7848>.

[RFC8334] Gould, J., Tan, W., and G. Brown, "Launch Phase Mapping for the Extensible Provisioning Protocol (EPP)", RFC 8334, DOI 10.17487/RFC8334, March 2018, <https://www.rfc-editor.org/info/rfc8334>.

9.2. Informative References

[RFC7451] Hollenbeck, S., "Extension Registry for the Extensible Provisioning Protocol", RFC 7451, DOI 10.17487/RFC7451, February 2015, <https://www.rfc-editor.org/info/rfc7451>.

9.3. URIs

[1] http://www.w3.org/TR/2004/REC-xmlschema-2-20041028/

Appendix A. Change History

A.1. Change from 00 to 01

1. Changed the XML namespace from urn:ietf:params:xml:ns:launchPolicy-0.1 to urn:ietf:params:xml:ns:epp:launchPolicy-0.1, and changed the XML schema registration from urn:ietf:params:xml:ns:launchPolicy-0.1 to urn:ietf:params:xml:schema:epp:launchPolicy-0.1 based on a request from IANA with draft-ietf-regext-allocation-token.

Author's Address

James Gould
VeriSign, Inc.
12061 Bluemont Way
Reston, VA 20190
US

Email: jgould@verisign.com
URI: http://www.verisigninc.com