Registration Protocols Extensions
Internet-Draft
Intended status: Standards Track
Expires: October 20, 2016

R. Carney
J. Snitker
GoDaddy Inc.

April 18, 2016

Validate Extension for the Extensible Provisioning Protocol (EPP)
draft-carney-regext-validate-00

Abstract

This document describes an Extensible Provisioning Protocol (EPP) extension mapping for the validation of contact and eligibility data.

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 October 20, 2016.

Copyright Notice

Table of Contents

1. Introduction

This document describes an extension mapping for version 1.0 of the Extensible Provisioning Protocol (EPP) [RFC5730]. This EPP mapping specifies a flexible schema by which EPP clients and servers can reliably validate contact and eligibility data.

With the increased number of restrictions on contacts and required data points (license, ids, etc.) to register a domain name, a way to validate the data points prior to issuing a transform command is becoming more important.

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.

2. Extension Elements

This extension provides a new EPP command called <validate>. As the set of EPP command verbs cannot be updated without updating the core EPP specifications, this command is implemented as an extension.

When a client wants to validate contact and/or eligibility data it will send an EPP command frame containing only an <extension> element. The <extension> element will contain a <validate> element. The intent with the complex <validate> element is to provide a mechanism to validate contacts and/or eligibility criteria to the policy level of a specific top level domain (TLD). The <validate:validate> element is used as a container of information to be validated.

This extension maintains a high reuse of current common EPP elements and attributes, though minor modifications and new elements/ attributes were required.

Due to different uses in this extension the following elements have been borrowed from RFC 5733 but have been redefined in the scope of the validate namespace: <validate:id>; <validate:postalInfo>; <validate:voice>; <validate:fax>; <validate:email>; <validate:authinfo>; <validate:disclose>; <validate:clTRID>

Key Value provides a flexible mechanism to share data between the client and the server. The <validate:kv> element defines the data, with two required simple attributes, key and value, and an optional contactType attribute for specificity in the response, more details below.

- An example <validate:kv key="VATID" value="0123456789"/>.
- An example <validate:kv contactType="Admin" key="contact:cc" value="Invalid country code for admin contact, must be MX."/>.

The <validate> element contains one child element:

- One or more <validate:contact> elements.

The <validate:contact> element has the following required attributes:

- One contactType attribute including but not limited to registrant, admin, tech, billing.
- One tld attribute.

The <validate:contact> element has the following required child elements:

- Zero or more <validate:kv> elements.
- One <validate:cd> element.

The <validate:cd> element has the following child elements:

- One <validate:id> element that contains the desired server-unique identifier for the contact to be created. This can be reused in other <validate:cd> elements in order to facilitate the reuse of the exact same data for other contact types.
- Zero to 2 <validate:postalInfo> elements that contain postal address information.
- An optional <validate:voice> element that contains the contact's voice telephone number.
- An optional <validate:fax> element that contains the contact's facsimile telephone number.
- An optional <validate:email> element that contains the contact's email address.
- An optional <validate:authInfo> element that contains authorization information to be associated with the contact object.
- An optional <validate:disclose> element that allows a client to identify elements that require exceptional server-operator handling to allow or restrict disclosure to third parties.

The following is an example of the <validate> extension command:

```
C:<?xml version="1.0" encoding="UTF-8" standalone="no"?>
C:<epp xmlns="urn:ietf:params:xml:ns:epp-1.0"
C:  xmlns:validate="urn:ietf:params:xml:ns:validate-0.1"
C:  xmlns:contact="urn:ietf:params:xml:ns:contact-1.0">
C:  <extension>
C:    <validate:validate>
C:      <validate:contact contactType="registrant" tld="COM">
C:        <validate:cd>
C:          <validate:id>sh8013</validate:id>
C:          <validate:postalInfo type="int">
C:            <contact:name>John Doe</contact:name>
C:            <contact:org>Example Inc.</contact:org>
C:            <contact:addr>
C:              <contact:street>123 Example Dr.</contact:street>
C:              <contact:street>Suite 100</contact:street>
C:              <contact:city>Dulles</contact:city>
C:              <contact:sp>VA</contact:sp>
C:              <contact:pc>20166-6503</contact:pc>
C:              <contact:cc>US</contact:cc>
C:            </contact:addr>
C:          </validate:postalInfo>
C:          <validate:voice>+1.7035555555</validate:voice>
```

```
C:          <validate:fax>+1.7035555556</validate:fax>
C:          <validate:email>jdoe@example.com</validate:email>
C:          <validate:authInfo>
C:            <contact:pw>2fooBAR</contact:pw>
C:          </validate:authInfo>
C:          <validate:disclose flag="0">
C:            <contact:voice/>
C:            <contact:email/>
C:          </validate:disclose>
C:        </validate:cd>
C:        <validate:kv key="VAT" value="1234567890"/>
C:      </validate:contact>
C:      <validate:contact contactType="tech" tld="COM">
C:        <validate:cd>
C:          <validate:id>sh8013</validate:id>
C:        </validate:cd>
C:      </validate:contact>
C:      <validate:contact contactType="admin" tld="COM">
C:        <validate:cd>
C:          <validate:id>sh8014</validate:id>
C:          <validate:postalInfo type="int">
C:            <contact:name>John Doe</contact:name>
C:            <contact:org>Example Inc.</contact:org>
C:            <contact:addr>
C:              <contact:street>123 Example Dr.</contact:street>
C:              <contact:street>Suite 100</contact:street>
C:              <contact:city>Dulles</contact:city>
C:              <contact:sp>VA</contact:sp>
C:              <contact:pc>20166-6503</contact:pc>
C:              <contact:cc>US</contact:cc>
C:            </contact:addr>
C:          </validate:postalInfo>
C:          <validate:voice>+1.7035555555</validate:voice>
C:          <validate:fax>+1.7035555556</validate:fax>
C:          <validate:email>jdoe@example.com</validate:email>
C:          <validate:authInfo>
C:            <contact:pw>2fooBAR</contact:pw>
C:          </validate:authInfo>
C:          <validate:disclose flag="0">
C:            <contact:voice/>
C:            <contact:email/>
C:          </validate:disclose>
C:        </validate:cd>
C:      </validate:contact>
C:      <validate:contact contactType="billing" tld="COM">
C:        <validate:cd>
C:          <validate:id>sh8014</validate:id>
C:        </validate:cd>
```

```
C:      </validate:contact>
C:    </validate:validate>
C:    <validate:clTRID>ABC-12345</validate:clTRID>
C:  </extension>
C:</epp>
```

If the server accepts a <validate> command, it MUST respond with a 1000. If the server rejects the command, it must respond with a 2400 result code.

```
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:    <extension>
S:      <validate:resData
S:        xmlns:validate="urn:ietf:params:xml:ns:validate-0.1">
S:        <validate:cd>
S:          <validate:id>sh8013</validate:id>
S:          <validate:response>1000</validate:response>
S:        </validate:cd>
S:        <validate:cd>
S:          <validate:id>sh8014</validate:id>
S:          <validate:response>2306</validate:response>
S:          <validate:kv key="contact:city" value="City not valid
S:            for state."/>
S:          <validate:kv contactType="Admin" key="contact:cc"
S:            value="Invalid country code for admin, must be mx."/>
S:          <validate:kv contactType="Billing" key="VAT" value="VAT
S:            required for Billing contact."/>
S:        </validate:cd>
S:      </validate:resData>
S:    </extension>
S:    <trID>
S:      <clTRID>ABC-12345</clTRID>
S:      <svTRID>54321-ZYX</svTRID>
S:    </trID>
S:  </response>
S:</epp>
```

3. Formal Syntax

One schema is presented here that is the EPP Validate Extension 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.

3.1. Validate Extension Schema

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

<schema
  targetNamespace="urn:ietf:params:xml:ns:validate-0.1"
  xmlns:validate="urn:ietf:params:xml:ns:validate-0.1"
  xmlns:epp="urn:ietf:params:xml:ns:epp-1.0"
  xmlns:eppcom="urn:ietf:params:xml:ns:eppcom-1.0"
  xmlns:domain="urn:ietf:params:xml:ns:domain-1.0"
  xmlns:contact="urn:ietf:params:xml:ns:contact-1.0"
  xmlns="http://www.w3.org/2001/XMLSchema"
  elementFormDefault="qualified">

  <annotation>
    <documentation>
      Extensible Provisioning Protocol v1.0 extension schema for
      command validate.
    </documentation>
  </annotation>

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

  <!-- Child elements found in validate commands. -->
  <element name="validate" type="validate:validateType"/>
  <element name="clTRID" type="epp:trIDStringType" />

  <complexType name="validateType">
    <sequence>
      <element name="contact" type="validate:validateContactType"
        maxOccurs="unbounded" />
    </sequence>
  </complexType>
```

```
  <complexType name="validateContactType">
    <sequence>
      <element name="cd" type="validate:createDataType"/>
      <element name="kv" type="validate:kvType" minOccurs="0"
      maxOccurs="unbounded" />
    </sequence>
    <attribute name="contactType" type="eppcom:labelType"
      use="required"/>
    <attribute name="tld" type="eppcom:labelType" use="required"/>
  </complexType>

  <complexType name="createDataType">
    <sequence>
      <element name="id" type="eppcom:clIDType" />
      <element name="postalInfo" type="contact:postalInfoType"
        minOccurs="0" maxOccurs="2" />
      <element name="voice" type="contact:e164Type" minOccurs="0" />
      <element name="fax" type="contact:e164Type" minOccurs="0" />
      <element name="email" type="eppcom:minTokenType" minOccurs="0"/>
      <element name="authInfo" type="contact:authInfoType"
        minOccurs="0"/>
      <element name="disclose" type="contact:discloseType"
        minOccurs="0" />
    </sequence>
  </complexType>

  <complexType name="kvType">
    <attribute name="contactType" type="eppcom:labelType"
      use="optional" />
    <attribute name="key" type="validate:keyType" use="required" />
    <attribute name="value" type="validate:valueType" use="required" />
  </complexType>

  <simpleType name="keyType">
    <restriction base="token">
      <minLength value="1" />
    </restriction>
  </simpleType>

  <simpleType name="valueType">
    <restriction base="token">
      <minLength value="0" />
    </restriction>
  </simpleType>

  <!-- Child response elements. -->
  <element name="resData" type="validate:resDataType" />
```

```
   <complexType name="resDataType">
     <sequence>
       <element name="cd" type="validate:resCreateDataType"
         maxOccurs="unbounded" />
     </sequence>
   </complexType>

   <complexType name="resCreateDataType">
     <sequence>
       <element name="id" type="eppcom:clIDType" />
       <element name="response" type="epp:resultCodeType" />
       <element name="kv" type="validate:kvType" minOccurs="0"
       maxOccurs="unbounded" />
     </sequence>
   </complexType>

 </schema>
 END
```

4. IANA Considerations

4.1. XML Namespace

This document uses URNs to describe XML namespaces and XML schemas conforming to a registry mechanism described in [RFC3688]. The following URI assignment is requested of IANA:

URI: ietf:params:xml:ns:validate-1.0

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

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

4.2. 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: "Validate Extension 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

5. 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.

6. Acknowledgements

The authors wish to thank the following persons for their feedback and suggestions:

- Kevin Allendorf of GoDaddy Inc.
- Jody Kolker of GoDaddy Inc.
- James Gould of Verisign Inc

7. Change History

8. 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, <http://www.rfc-editor.org/info/rfc2119>.

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

[RFC5730] Hollenbeck, S., "Extensible Provisioning Protocol (EPP)", STD 69, RFC 5730, DOI 10.17487/RFC5730, August 2009, <http://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, <http://www.rfc-editor.org/info/rfc5731>.

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

Authors' Addresses

Roger Carney
GoDaddy Inc.
14455 N. Hayden Rd. #219
Scottsdale, AZ 85260
US

Email: rcarney@godaddy.com
URI: http://www.godaddy.com

Joseph Snitker
GoDaddy Inc.
14455 N. Hayden Rd. #219
Scottsdale, AZ 85260
US

Email: jsnitker@godaddy.com
URI: http://www.godaddy.com