Internet Engineering Task Force (IETF)
Internet-Draft
Intended status: Standards Track
Expires: January 3, 2022

T. Sattler

J. Kolker
GoDaddy Inc.

December 21, 2021

Object Search for the Extensible Provisioning Protocol (EPP)
draft-sattler-kolker-epp-object-search-00

Abstract

This document describes an Extensible Provisioning Protocol (EPP) extension called "Object Search", which EPP clients use to search and list EPP objects stored by EPP servers.

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 https://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 January 3, 2022.

Table of Contents

1. Introduction

The Extensible Provisioning Protocol (EPP), as defined in [RFC5730], is a protocol whose original motivation is to provide a standard Internet domain name registration protocol for use between registries and registrars.

Registrars can transform EPP objects either with <create>, <delete>, <renew>, <transfer>, or <update> according to [RFC5730]. They are usually responsible for maintaining their inventory themselves.

Due to various reasons, such as consolidation of registrars, migrations, development of new management systems, etc., it may happen that a registrar has lost track of its inventory.

EPP already provides query capabilities with <check> and <info>, but those are limited to looking up exact strings. Therefore, it is desirable to implement a search functionality via EPP to find any forgotten or orphaned objects.

This document describes an extension mapping for version 1.0 of the EPP to provide a mechanism by which EPP clients can search and list EPP objects stored by EPP servers.

1.1. Terminology and Definitions

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "NOT RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in BCP 14 [RFC2119] [RFC8174] when, and only when, they appear in all capitals, as shown here.

XML [W3C.REC-xml11-20060816] is case-sensitive. Unless stated otherwise, XML specification and examples provided in this document MUST be interpreted in the character case presented in order to develop a conforming implementation.

"object-search" is used as an abbreviation for "urn:ietf:params:xml:ns:epp:object-search-0.1". The XML namespace prefix "object-search" is used, but implementations MUST NOT depend on it. Instead, they are to employ a proper namespace-aware XML parser and serializer to interpret and output the XML documents.

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. Migrating to Newer Versions of This Extension

Servers that implement this extension SHOULD provide a way for clients to progressively update their implementations when a new version of the extension is deployed. A newer version of the extension is expected to use an XML namespace with a higher version number than the prior versions.

Servers SHOULD (for a temporary migration period up to server policy) provide support for older versions of the extension in parallel to the newest version and allow clients to execute their preferred version of the <info> command based on the object search <objURI> elements of the server <greeting>. The version of the object search <info> response MUST match the version of the object search <info> command executed by the server.

3. Object Attributes

3.1. Object Search Elements

The minimum character for a search pattern and the the limit for the search results is up to server policy.

<object-search:domain>
The search pattern for the domain object; the attribute "type" is REQUIRED and MUST either be "name", "contact", "host", "status", "crDate", "upDate", "exDate", or "trDate". The attribute "filter" REQUIRED and MUST be present to identify the filter and MUST either be "exact", "begins", "ends", or "contains".

<object-search:host>
The search pattern for the host object; the attribute "type" is REQUIRED and MUST either be "name", "ipv4", or "ipv6". The attribute "filter" is REQUIRED and MUST be present to identify the filter and MUST either be "exact", "begins", "ends", or "contains".

<object-search:contact>
The search pattern for the contact object; the attribute "type" is REQUIRED and MUST either be "id", "name", "orga", "street", "city", "state", "postcode", "email", "voice", "fax", or "country". The attribute "filter" is REQUIRED and MUST be present to identify the filter and MUST either be "exact", "begins", "ends", or "contains".

4. 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 used to search for objects and object mapping.

4.1. EPP Query Commands

EPP [RFC5730] provides four 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, <poll> to discover and retrieve service messages, and <transfer> to retrieve object transfer status information.

This extension does not add any elements to EPP <check>, <poll>, and <transfer> commands or responses.

4.1.1. EPP <info> Command

EPP provides the <info> command that is used to looking up objects. In addition to the standard EPP command elements, the <info> command MUST contain a <object-search:info> element that allows objects to be searched.

The <object-search:info> element MUST contain a child element. It is either the <object-search:domain> child element, described in Section 4.1.1.1, to search domains, the <object-search:contact> child element, described in Section 4.1.1.2, to search contacts, or the <object-search:host> child element, described in Section 4.1.1.3, to search hosts.

4.1.1.1. Object Search Domains

A search for domain objects can be performed by using the <info> command with the <object-search> element and the <object-search:domain> child element, defined in Section 3.1.

Example to search for a domain object in an <info> command.

```
C:<?xml version="1.0" encoding="UTF-8" standalone="no"?>
C:<epp xmlns="urn:ietf:params:xml:ns:epp-1.0">
C:  <command>
C:    <info>
C:      <object-search:info xmlns:object-search=
C:        "urn:ietf:params:xml:ns:epp:object-search-0.1">
C:        <object-search:domain type="name" filter="begins">example
C:        </object-search:domain>
C:      </object-search: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 <object-search:infData> element that identifies the object search namespace. The <object-search:infData> element contains the <object-search:list> element with zero or more <object-search:item> child elements. The "total" attribute is REQUIRED and MUST contain the total number of the search result. This allows the client to determine if the search needs to be refined if the number is too large. The <object-search:item> element contains <object-search:domain> child elements defined in Section 3.1.

Example of returning the list of object search items in an <info> response.

```
S:<?xml version="1.0" encoding="UTF-8"?>
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:        <object-search:infData xmlns:object-search=
S:          "urn:ietf:params:xml:ns:epp:object-search-0.1">
S:          <object-search:list total="2">
S:            <object-search:item>
S:              <object-search:domain type="name" filter="begins">
S:                example.com
S:              </object-search:domain>
S:            </object-search:item>
S:            <object-search:item>
S:              <object-search:domain type="name" filter="begins">
S:                example.net
S:              </object-search:domain>
S:            </object-search:item>
S:          </object-search:list>
S:        </object-search:infData>
S:      </resData>
S:      <trID>
S:         <clTRID>ABC-12345</clTRID>
S:         <svTRID>54321-XYZ</svTRID>
S:      </trID>
S:   </response>
S:</epp>
```

4.1.1.2. Object Search Contacts

A search for contact objects can be performed by using the <info> command with the <object-search> element and the <object-search:contact> child element, defined in Section 3.1.

Example to search for a contact object in an <info> command.

```
C:<?xml version="1.0" encoding="UTF-8" standalone="no"?>
C:<epp xmlns="urn:ietf:params:xml:ns:epp-1.0">
C:  <command>
C:    <info>
C:      <object-search:info xmlns:object-search=
C:        "urn:ietf:params:xml:ns:epp:object-search-0.1">
C:        <object-search:contact type="name" filter="contains">John
C:        </object-search:contact>
C:      </object-search:info>
C:    </info>
C:    <clTRID>ABC-12346</clTRID>
C:  </command>
C:</epp>
```

When an <info> command has been processed successfully, the EPP <resData> element MUST contain a child <object-search:infData> element that identifies the object search namespace. The <object-search:infData> element contains the <object-search:list> element with zero or more <object-search:item> child elements. The "total" attribute is REQUIRED and MUST contain the total number of the search result. This allows the client to determine if the search needs to be refined if the number is too large. The <object-search:item> element contains the <object-search:contact> child element defined in Section 3.1.

Example of returning the list of object search items in an <info> response.

```
S:<?xml version="1.0" encoding="UTF-8"?>
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:        <object-search:infData xmlns:object-search=
S:          "urn:ietf:params:xml:ns:epp:object-search-0.1">
S:          <object-search:list total="12345678">
S:            <object-search:item>
S:              <object-search:contact type="name" filter="contains">
S:                sh8013
S:              </object-search:contact>
S:            </object-search:item>
S:            <object-search:item>
S:              <object-search:contact type="name" filter="contains">
S:                sah8013
S:              </object-search:contact>
S:            </object-search:item>
S:          </object-search:list>
S:        </object-search:infData>
S:      </resData>
S:      <trID>
S:         <clTRID>ABC-12346</clTRID>
S:         <svTRID>54322-XYZ</svTRID>
S:      </trID>
S:   </response>
S:</epp>
```

4.1.1.3 Object Search Hosts

A search for host objects can be performed by using the <info> command with the <object-search> element and the <object-search:host> child element, defined in Section 3.1.

Example to search for a host object in an <info> command.

```
C:<?xml version="1.0" encoding="UTF-8" standalone="no"?>
C:<epp xmlns="urn:ietf:params:xml:ns:epp-1.0">
C:  <command>
C:    <info>
C:      <object-search:info xmlns:object-search=
C:        "urn:ietf:params:xml:ns:epp:object-search-0.1">
C:        <object-search:host type="name" filter="ends">example.com
C:        </object-search:host>
```

```
C:      </object-search:info>
C:    </info>
C:    <clTRID>ABC-12347</clTRID>
C:  </command>
C:</epp>
```

When an <info> command has been processed successfully, the EPP <resData> element MUST contain a child <object-search:infData> element that identifies the object search namespace. The <object-search:infData> element contains the <object-search:list> element with zero or more <object-search:item> child elements. The "total" attribute is REQUIRED and MUST contain the total number of the search result. This allows the client to determine if the search needs to be refined if the number is too large. The <object-search:item> element contains the <object-search:host> child element defined in Section 3.1.

Example of returning the list of object search items in an <info> response.

```
S:<?xml version="1.0" encoding="UTF-8"?>
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:        <object-search:infData xmlns:object-search=
S:          "urn:ietf:params:xml:ns:epp:object-search-0.1">
S:          <object-search:list total="34567">
S:            <object-search:item>
S:              <object-search:host type="name" filter="ends">
S:                ns1.example.com
S:              </object-search:host>
S:            </object-search:item>
S:            <object-search:item>
S:              <object-search:host type="name" filter="ends">
S:                ns2.example.com
S:              </object-search:host>
S:            </object-search:item>
S:          </object-search:list>
S:        </object-search:infData>
S:      </resData>
S:      <trID>
S:         <clTRID>ABC-12347</clTRID>
S:         <svTRID>54323-XYZ</svTRID>
S:      </trID>
S:   </response>
S:</epp>
```

4.2. 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.

This extension does not add any elements to the EPP <create>, <delete>, <renew>, <transfer>, and <update>.

5. Formal Syntax

The EPP Object Search schema is presented here.

The formal syntax presented here is a complete schema representation of the object mapping suitable for automated validation of EPP XML instances. The <CODE BEGINS> and <CODE ENDS> tags are not part of the schema; they are used to note the beginning and end of the schema for URI registration purposes.

5.1. Object Search EPP Mapping Schema

```
<CODE BEGINS>
<?xml version="1.0" encoding="UTF-8"?>
  <schema targetNamespace=
    "urn:ietf:params:xml:ns:epp:object-search-0.1"
    xmlns:eppcom="urn:ietf:params:xml:ns:eppcom-1.0"
    xmlns:epp="urn:ietf:params:xml:ns:epp-1.0"
    xmlns:object-search=
      "urn:ietf:params:xml:ns:epp:object-search-0.1"
    xmlns="https://www.w3.org/2001/XMLSchema"
    elementFormDefault="qualified">
  <!--
  Import common element types
  -->
  <import namespace="urn:ietf:params:xml:ns:eppcom-1.0"/>
  <import namespace="urn:ietf:params:xml:ns:epp-1.0"/>
  <annotation>
    <documentation>
      Extensible Provisioning Protocol v1.0
      Object Search Mapping Schema.
    </documentation>
  </annotation>
  <!--
   Info Response element
  -->
  <element name="infData" type="object-search:infDataType"/>
  <!--
```

```
      <info> response elements.
     -->
     <complexType name="infDataType">
       <sequence>
```

```
        <element name="list" type="object-search:listDataType"/>
      </sequence>
    </complexType>
    <!--
     Attributes associated with the list info response
    -->
    <complexType name="listDataType">
      <choice>
        <element name="item" type="object-search:domainItemType"
         minOccurs="0" maxOccurs="unbounded"/>
        <element name="item" type="object-search:contactItemType"
         minOccurs="0" maxOccurs="unbounded"/>
        <element name="item" type="object-search:hostItemType"
         minOccurs="0" maxOccurs="unbounded"/>
      </choice>
    </complexType>
    <!--
     Attributes associated with the list item info response
    -->
    <complexType name="domainItemType">
      <sequence>
        <element name="type" type="object-search:domainTypeEnum"/>
        <element name="filter" type="object-search:filterEnum"/>
      </sequence>
    </complexType>
    <!--
     Attributes associated with the list item info response
    -->
    <complexType name="contactItemType">
      <sequence>
        <element name="type" type="object-search:contactTypeEnum"/>
        <element name="filter" type="object-search:filterEnum"/>
      </sequence>
    </complexType>
    <!--
     Attributes associated with the list item info response
    -->
    <complexType name="hostItemType">
      <sequence>
        <element name="type" type="object-search:hostTypeEnum"/>
        <element name="filter" type="object-search:filterEnum"/>
      </sequence>
    </complexType>
    <!--
     Enumerated list of domain types
    -->
    <simpleType name="domainTypeEnum">
      <restriction base="token">
        <enumeration value="name"/>
```

```
          <enumeration value="contact"/>
          <enumeration value="host"/>
          <enumeration value="status"/>
```

```
        <enumeration value="crDate"/>
        <enumeration value="upDate"/>
        <enumeration value="exDate"/>
        <enumeration value="trDate"/>
      </restriction>
    </simpleType>
    <!--
     Enumerated list of contact types
    -->
    <simpleType name="contactTypeEnum">
      <restriction base="token">
        <enumeration value="id"/>
        <enumeration value="name"/>
        <enumeration value="orga"/>
        <enumeration value="street"/>
        <enumeration value="city"/>
        <enumeration value="state"/>
        <enumeration value="postcode"/>
        <enumeration value="email"/>
        <enumeration value="voice"/>
        <enumeration value="fax"/>
        <enumeration value="country"/>
      </restriction>
    </simpleType>
    <!--
     Enumerated list of host types
    -->
    <simpleType name="hostTypeEnum">
      <restriction base="token">
        <enumeration value="name"/>
        <enumeration value="ipv4"/>
        <enumeration value="ipv6"/>
      </restriction>
    </simpleType>
    <!--
     Enumerated list of filters
    -->
    <simpleType name="filterEnum">
      <restriction base="token">
        <enumeration value="exact"/>
        <enumeration value="begins"/>
        <enumeration value="ends"/>
        <enumeration value="contains"/>
      </restriction>
    </simpleType>
    <!--
     End of schema.
    -->
    </schema>
```

<CODE ENDS>

6. IANA Considerations

6.1. XML Namespace

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

Registration request for the object search namespace:
URI: urn:ietf:params:xml:ns:epp:object-search-0.1
Registrant Contact: IESG
XML: None. Namespace URIs do not represent an XML specification.

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

6.2. EPP Extension Registry

The following registration of the EPP Extension Registry, described in [RFC7451], is requested:

Name of Extension: Object Search for the Extensible Provisioning Protocol (EPP)
Document status: Standards Track
Reference: (insert the 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

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. Implementation Status

Note to RFC Editor: Please remove this section and the reference to [RFC7942] before publication.

This section records the status of known implementations of the protocol defined by this specification at the time of posting of this Internet-Draft, and is based on a proposal described in [RFC7942]. The description of implementations in this section is intended to assist the IETF in its decision processes in progressing drafts to RFCs. Please note that the listing of any individual implementation here does not imply endorsement by the IETF. Furthermore, no effort has been spent to verify the information presented here that was supplied by IETF contributors. This is not intended as, and must not be construed to be, a catalog of available implementations or their features. Readers are advised to note that other implementations may exist.

According to [RFC7942], "this will allow reviewers and working groups to assign due consideration to documents that have the benefit of running code, which may serve as evidence of valuable experimentation and feedback that have made the implemented protocols more mature. It is up to the individual working groups to use this information as they see fit".

9. References

9.1. Normative References

[W3C.REC-xml11-20060816]
Bray, T., Paoli, J., Sperberg-McQueen, M., Maler, E., Yergeau, F., and J. Cowan, "Extensible Markup Language (XML) 1.1 (Second Edition)", World Wide Web Consortium Recommendation REC-xml11-20060816, 16 August 2006, <https://www.w3.org/TR/2006/REC-xml11-20060816>.

Latest version available at <https://www.w3.org/TR/xml11/>.

[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>.

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

[RFC7942] Sheffer, Y. and A. Farrel, "Improving Awareness of Running Code: The Implementation Status Section", BCP 205,

RFC 7942, DOI 10.17487/RFC7942, July 2016,
<https://www.rfc-editor.org/info/rfc7942>.

[RFC8174] Leiba, B., "Ambiguity of Uppercase vs Lowercase in RFC 2119 Key Words", BCP 14, RFC 8174, DOI 10.17487/RFC8174, May 2017, <https://www.rfc-editor.org/info/rfc8174>.

9.2. Informative References

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

[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>.

Appendix A. Change History

TBD

Acknowledgments

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

Authors' Addresses

Tobias Sattler

Email: mail@tobiassattler.com
URI: https://tobiassattler.com

Jody Kolker
GoDaddy Inc.
2155 E GoDaddy Way
Tempe, AZ 85284
United States of America

Email: jkolker@godaddy.com
URI: https://www.godaddy.com