RFC 9038
Extensible Provisioning Protocol (EPP) Unhandled Namespaces

Abstract

The Extensible Provisioning Protocol (EPP), as defined in RFC 5730, includes a method for the client and server to determine the objects to be managed during a session and the object extensions to be used during a session. The services are identified using namespace URIs, and an "unhandled namespace" is one that is associated with a service not supported by the client. This document defines an operational practice that enables the server to return information associated with unhandled namespace URIs and that maintains compliance with the negotiated services defined in RFC 5730.

Status of This Memo

This is an Internet Standards Track document.

This document is a product of the Internet Engineering Task Force (IETF). It represents the consensus of the IETF community. It has received public review and has been approved for publication by the Internet Engineering Steering Group (IESG). Further information on Internet Standards is available in Section 2 of RFC 7841.

Information about the current status of this document, any errata, and how to provide feedback on it may be obtained at https://www.rfc-editor.org/info/rfc9038.

Copyright Notice

Table of Contents

1. Introduction

The Extensible Provisioning Protocol (EPP), as defined in [RFC5730], includes a method for the client and server to determine the objects to be managed during a session and the object extensions to be used during a session. The services are identified using namespace URIs. How should the server handle service data that needs to be returned in the response when the client does not support the required service namespace URI, which is referred to as an "unhandled namespace"? An unhandled namespace is a significant issue for the processing of the poll messages described in [RFC5730], since poll messages are inserted by the server prior to knowing the supported client services, and the client needs to be capable of processing all poll messages. Returning an unhandled namespace poll message is not compliant with the negotiated services defined in [RFC5730], and returning an error makes the unhandled namespace poll message a poison message by halting the processing of the poll queue. An unhandled namespace is also an issue for general EPP responses when the server has information that it cannot return to the client due to the client's supported services. The server should be able to return unhandled namespace information that the client can process later. This document defines an operational practice that enables the server to return information associated with unhandled namespace URIs and that maintains compliance with the negotiated services defined in [RFC5730].

1.1. Conventions Used in This Document

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 specifications and examples provided in this document **MUST** be interpreted in the character case presented in order to develop a conforming implementation.

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

The examples reference XML namespace prefixes that are used for the associated XML namespaces. Implementations **MUST NOT** depend on the example XML namespaces and instead employ a proper namespace-aware XML parser and serializer to interpret and output the XML documents. The example namespace prefixes used and their associated XML namespaces include:

changePoll: urn:ietf:params:xml:ns:changePoll-1.0

domain: urn:ietf:params:xml:ns:domain-1.0

secDNS: urn:ietf:params:xml:ns:secDNS-1.1

In the template example XML, placeholder content is represented by the following variables:

[NAMESPACE-XML]:   XML content associated with a login service namespace URI. An example is the <domain:infData> element content in [RFC5731].

[NAMESPACE-URI]:   XML namespace URI associated with the [NAMESPACE-XML] XML content. An example is "urn:ietf:params:xml:ns:domain-1.0" in [RFC5731].

2. Unhandled Namespaces

An unhandled namespace is an XML namespace that is associated with a response extension that is not included in the client-specified EPP login services of [RFC5730]. The EPP login services consist of the set of XML namespace URIs included in the <objURI> or <extURI> elements of the EPP <login> command [RFC5730]. The services supported by the server are included in the <objURI> and <extURI> elements of the EPP <greeting> [RFC5730], which should be a superset of the login services included in the EPP <login> command. A server may have information associated with a specific namespace that it needs to return in the response to a client. The unhandled namespaces problem exists when the server has information that it needs to return to the client, but the namespace of the information is not supported by the client based on the negotiated EPP <login> command services.

3. Use of EPP <extValue> for Unhandled Namespace Data

In [RFC5730], the <extValue> element is used to provide additional error diagnostic information, including the <value> element that identifies the client-provided element that caused a server error condition and the <reason> element containing the human-readable message that describes the reason for the error. This operational practice extends the use of the <extValue> element for the purpose of returning unhandled namespace information in a successful response.

When a server has data to return to the client that the client does not support based on the login services, the server **MAY** return a successful response with the data for each unsupported namespace moved into an <extValue> element [RFC5730]. The unhandled namespace will not cause an error response, but the unhandled namespace data will instead be moved to an <extValue> element, along with a reason why the unhandled namespace data could not be included in the appropriate location of the response. The <extValue> element will not be processed by the XML processor. The <extValue> element contains the following child elements:

<value>:   Contains a child element with the unhandled namespace XML. The unhandled namespace **MUST** be declared in the child element or any containing element, including the root element. XML processing of the <value> element is disabled by the XML schema in [RFC5730], so the information can safely be returned in the <value> element.

<reason>:   A formatted, human-readable message that indicates the reason the unhandled namespace data was not returned in the appropriate location of the response. The formatted reason **SHOULD** follow the Augmented Backus-Naur Form (ABNF) grammar [RFC5234] format: NAMESPACE-URI " not in login services", where NAMESPACE-URI is the unhandled XML namespace like "urn:ietf:params:xml:ns:domain-1.0" in [RFC5731].

This document applies to the handling of unsupported namespaces for object-level extensions and command-response extensions [RFC3735]. This document does not apply to the handling of unsupported namespaces for protocol-level extensions or authentication-information extensions [RFC3735]. Refer to the following sections on how to handle an unsupported object-level extension namespace or an unsupported command-response extension namespace.

3.1. Unhandled Object-Level Extension

An object-level extension in [RFC5730] is a child element of the <resData> element. If the client does not handle the namespace of the object-level extension, then the <resData> element is removed and its object-level extension child element is moved into an <extValue> <value> element [RFC5730], with the namespace URI included in the corresponding <extValue> <reason> element. The response becomes a general EPP response without the <resData> element.

Below is a template response for a supported object-level extension. The [NAMESPACE-XML] variable represents the object-level extension XML.

```
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:      [NAMESPACE-XML]
S:    </resData>
S:    <trID>
S:      <clTRID>ABC-12345</clTRID>
S:      <svTRID>54322-XYZ</svTRID>
S:    </trID>
S:  </response>
S:</epp>
```

Below is a template for an unhandled namespace response for an unsupported object-level extension. The [NAMESPACE-XML] variable represents the object-level extension XML, and the [NAMESPACE-URI] variable represents the object-level extension XML namespace URI.

```
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:      <extValue>
S:        <value>
S:          [NAMESPACE-XML]
S:        </value>
S:        <reason>
S:          [NAMESPACE-URI] not in login services
S:        </reason>
S:      </extValue>
S:    </result>
S:    <trID>
S:      <clTRID>ABC-12345</clTRID>
S:      <svTRID>54322-XYZ</svTRID>
S:    </trID>
S:  </response>
S:</epp>
```

The EPP response is converted from an object response to a general EPP response by the server when the client does not support the object-level extension namespace URI.

Below is an example of a <transfer> query response (see Section 3.1.3 of [RFC5731]) converted into an unhandled namespace 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:      <extValue>
S:        <value>
S:          <domain:trnData
S:            xmlns:domain="urn:ietf:params:xml:ns:domain-1.0">
S:            <domain:name>example.com</domain:name>
S:            <domain:trStatus>pending</domain:trStatus>
S:            <domain:reID>ClientX</domain:reID>
S:            <domain:reDate>2000-06-06T22:00:00.0Z</domain:reDate>
S:            <domain:acID>ClientY</domain:acID>
S:            <domain:acDate>2000-06-11T22:00:00.0Z</domain:acDate>
S:            <domain:exDate>2002-09-08T22:00:00.0Z</domain:exDate>
S:          </domain:trnData>
S:        </value>
S:        <reason>
S:          urn:ietf:params:xml:ns:domain-1.0 not in login services
S:        </reason>
S:      </extValue>
S:    </result>
S:    <trID>
S:      <clTRID>ABC-12345</clTRID>
S:      <svTRID>54322-XYZ</svTRID>
S:    </trID>
S:  </response>
S:</epp>
```

3.2. Unhandled Command-Response Extension

A command-response extension in [RFC5730] is a child element of the <extension> element. If the client does not handle the namespace of the command-response extension, the command-response child element is moved into an <extValue> <value> element [RFC5730], with the namespace URI included in the corresponding <extValue> <reason> element. Afterwards, if there are no additional command-response child elements, the <extension> element **MUST** be removed.

Below is a template response for a supported command-response extension. The [NAMESPACE-XML] variable represents the command-response extension XML.

```
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:      [NAMESPACE-XML]
S:    </extension>
S:    <trID>
S:      <clTRID>ABC-12345</clTRID>
S:      <svTRID>54322-XYZ</svTRID>
S:    </trID>
S:  </response>
S:</epp>
```

Below is a template of an unhandled namespace response for an unsupported command-response extension. The [NAMESPACE-XML] variable represents the command-response extension XML, and the [NAMESPACE-URI] variable represents the command-response extension XML namespace URI.

```
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:      <extValue>
S:        <value>
S:         [NAMESPACE-XML]
S:        </value>
S:        <reason>
S:          [NAMESPACE-URI] not in login services
S:        </reason>
S:      </extValue>
S:    </result>
S:    <trID>
S:      <clTRID>ABC-12345</clTRID>
S:      <svTRID>54322-XYZ</svTRID>
S:    </trID>
S:  </response>
S:</epp>
```

The EPP response is converted to an unhandled namespace response by moving the unhandled command-response extension from under the <extension> to an <extValue> element.

Below is example of the Delegation Signer (DS) Data Interface <info> response (see Section 5.1.2 of [RFC5910]) converted to an unhandled namespace 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:      <extValue>
S:        <value>
S:          <secDNS:infData
S:            xmlns:secDNS="urn:ietf:params:xml:ns:secDNS-1.1">
S:            <secDNS:dsData>
S:              <secDNS:keyTag>12345</secDNS:keyTag>
S:              <secDNS:alg>3</secDNS:alg>
S:              <secDNS:digestType>1</secDNS:digestType>
S:              <secDNS:digest>49FD46E6C4B45C55D4AC</secDNS:digest>
S:            </secDNS:dsData>
S:          </secDNS:infData>
S:        </value>
S:        <reason>
S:          urn:ietf:params:xml:ns:secDNS-1.1 not in login services
S:        </reason>
S:      </extValue>
S:    </result>
S:    <resData>
S:      <domain:infData
S:        xmlns:domain="urn:ietf:params:xml:ns:domain-1.0">
S:        <domain:name>example.com</domain:name>
S:        <domain:roid>EXAMPLE1-REP</domain:roid>
S:        <domain:status s="ok"/>
S:        <domain:registrant>jd1234</domain:registrant>
S:        <domain:contact type="admin">sh8013</domain:contact>
S:        <domain:contact type="tech">sh8013</domain:contact>
S:        <domain:ns>
S:          <domain:hostObj>ns1.example.com</domain:hostObj>
S:          <domain:hostObj>ns2.example.com</domain:hostObj>
S:        </domain:ns>
S:        <domain:host>ns1.example.com</domain:host>
S:        <domain:host>ns2.example.com</domain:host>
S:        <domain:clID>ClientX</domain:clID>
S:        <domain:crID>ClientY</domain:crID>
S:        <domain:crDate>1999-04-03T22:00:00.0Z</domain:crDate>
S:        <domain:upID>ClientX</domain:upID>
S:        <domain:upDate>1999-12-03T09:00:00.0Z</domain:upDate>
S:        <domain:exDate>2005-04-03T22:00:00.0Z</domain:exDate>
S:        <domain:trDate>2000-04-08T09:00:00.0Z</domain:trDate>
S:        <domain:authInfo>
S:          <domain:pw>2fooBAR</domain:pw>
S:        </domain:authInfo>
S:      </domain:infData>
S:    </resData>
S:    <trID>
S:      <clTRID>ABC-12345</clTRID>
S:      <svTRID>54322-XYZ</svTRID>
S:    </trID>
S:  </response>
S:</epp>
```

4. Signaling Client and Server Support

This document does not define new EPP protocol elements but rather specifies an operational practice using the existing EPP protocol, where the client and the server can signal support for the operational practice using a namespace URI in the login and greeting extension services. The namespace URI "urn:ietf:params:xml:ns:epp:unhandled-namespaces-1.0" is used to signal support for the operational practice. The client includes the namespace URI in an <svcExtension> <extURI> element of the <login> command [RFC5730]. The server includes the namespace URI in an <svcExtension> <extURI> element of the greeting [RFC5730].

A client that receives the namespace URI in the server's greeting extension services can expect the following supported behavior by the server:

- support unhandled namespace object-level extensions and command-response extensions in EPP poll messages, per Section 6
- support the option of unhandled namespace command-response extensions in general EPP responses, per Section 5

A server that receives the namespace URI in the client's <login> command extension services can expect the following supported behavior by the client:

- support monitoring the EPP poll messages and general EPP responses for unhandled namespaces

5. Usage with General EPP Responses

The unhandled namespace approach defined in Section 3 **MAY** be used for a general EPP response to an EPP command. A general EPP response includes any EPP response that is not a poll message. The use of the unhandled namespace approach for poll-message EPP responses is defined in Section 6. The server **MAY** exclude the unhandled namespace information in the general EPP response or **MAY** include it using the unhandled namespace approach.

The unhandled namespace approach for general EPP responses **SHOULD** only be applicable to command-response extensions, defined in Section 3.2, since the server **SHOULD NOT** accept an object-level EPP command if the client did not include the object-level namespace URI in the login services. An object-level EPP response extension is returned when the server successfully executes an object-level EPP command extension. The server **MAY** return an unhandled object-level extension to the client, as defined in Section 3.1.

Returning domain name Redemption Grace Period (RGP) data, based on [RFC3915], provides an example of applying the unhandled namespace approach for a general EPP response. If the client does not include the "urn:ietf:params:xml:ns:rgp-1.0" namespace URI in the login services and the domain <info> response of a domain name does have RGP information, the server **MAY** exclude the <rgp:infData> element from the EPP response or **MAY** include it under the <extValue> element, per Section 3.2.

Below is an example of a domain name <info> response [RFC5731] converted to an unhandled
<rgp:infData> element (see Section 4.1.1 of [RFC3915]) included under an <extValue> element:

```
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:     xsi:schemaLocation="urn:ietf:params:xml:ns:epp-1.0
S:     epp-1.0.xsd">
S:  <response>
S:    <result code="1000">
S:      <msg>Command completed successfully</msg>
S:      <extValue>
S:        <value>
S:          <rgp:infData xmlns:rgp="urn:ietf:params:xml:ns:rgp-1.0"
S:           xsi:schemaLocation="urn:ietf:params:xml:ns:rgp-1.0
S:           rgp-1.0.xsd">
S:            <rgp:rgpStatus s="redemptionPeriod"/>
S:          </rgp:infData>
S:        </value>
S:        <reason>
S:          urn:ietf:params:xml:ns:rgp-1.0 not in login services
S:        </reason>
S:      </extValue>
S:    </result>
S:    <resData>
S:      <domain:infData
S:        xmlns:domain="urn:ietf:params:xml:ns:domain-1.0"
S:        xsi:schemaLocation="urn:ietf:params:xml:ns:domain-1.0
S:        domain-1.0.xsd">
S:        <domain:name>example.com</domain:name>
S:        <domain:roid>EXAMPLE1-REP</domain:roid>
S:        <domain:status s="pendingDelete"/>
S:        <domain:registrant>jd1234</domain:registrant>
S:        <domain:contact type="admin">sh8013</domain:contact>
S:        <domain:contact type="tech">sh8013</domain:contact>
S:        <domain:ns>
S:          <domain:hostObj>ns1.example.com</domain:hostObj>
S:          <domain:hostObj>ns1.example.net</domain:hostObj>
S:        </domain:ns>
S:        <domain:host>ns1.example.com</domain:host>
S:        <domain:host>ns2.example.com</domain:host>
S:        <domain:clID>ClientX</domain:clID>
S:        <domain:crID>ClientY</domain:crID>
S:        <domain:crDate>1999-04-03T22:00:00.0Z</domain:crDate>
S:        <domain:upID>ClientX</domain:upID>
S:        <domain:upDate>1999-12-03T09:00:00.0Z</domain:upDate>
S:        <domain:exDate>2005-04-03T22:00:00.0Z</domain:exDate>
S:        <domain:trDate>2000-04-08T09:00:00.0Z</domain:trDate>
S:        <domain:authInfo>
S:          <domain:pw>2fooBAR</domain:pw>
S:        </domain:authInfo>
S:      </domain:infData>
S:    </resData>
S:    <trID>
S:      <clTRID>ABC-12345</clTRID>
S:      <svTRID>54322-XYZ</svTRID>
S:    </trID>
S:  </response>
S:</epp>
```

6. Usage with Poll-Message EPP Responses

The unhandled namespace approach, defined in Section 3, MUST be used if there is unhandled namespace information included in a <poll> response. The server inserts poll messages into the client's poll queue independent of knowing the supported client login services; therefore, there may be unhandled object-level extensions and command-response extensions included in a client's poll queue. In [RFC5730], the <poll> command is used by the client to retrieve and acknowledge poll messages that have been inserted by the server. The <poll> response is an EPP response that includes the <msgQ> element that provides poll queue metadata about the message. The unhandled namespace approach, defined in Section 3, is used for an unhandled object-level extension and for each of the unhandled command-response extensions attached to the <poll> response. The resulting <poll> response MAY have either or both the object-level extension or command-response extensions moved to <extValue> elements, as defined in Section 3.

The change poll message, as defined in Section 3.1.2 of [RFC8590], which is an extension of any EPP object, is an example of applying the unhandled namespace approach for <poll> responses. Below are examples of converting the domain name <info> response example in Section 3.1.2 of [RFC8590] to an unhandled namespace response. The object that will be used in the examples is a domain name object [RFC5731].

Below is a domain name <info> <poll> response [RFC5731] with the unhandled
<changePoll:changeData> element [RFC8590] included under an <extValue> element.

```
S:<?xml version="1.0" encoding="UTF-8" standalone="no"?>
S:<epp xmlns="urn:ietf:params:xml:ns:epp-1.0">
S:  <response>
S:    <result code="1301">
S:      <msg lang="en-US">
S:        Command completed successfully; ack to dequeue</msg>
S:      <extValue>
S:        <value>
S:          <changePoll:changeData
S:           xmlns:changePoll="urn:ietf:params:xml:ns:changePoll-1.0"
S:           state="after">
S:            <changePoll:operation>update</changePoll:operation>
S:            <changePoll:date>
S:              2013-10-22T14:25:57.0Z</changePoll:date>
S:            <changePoll:svTRID>12345-XYZ</changePoll:svTRID>
S:            <changePoll:who>URS Admin</changePoll:who>
S:            <changePoll:caseId type="urs">urs123
S:            </changePoll:caseId>
S:            <changePoll:reason>URS Lock</changePoll:reason>
S:          </changePoll:changeData>
S:        </value>
S:        <reason>
S:        urn:ietf:params:xml:ns:changePoll-1.0 not in login services
S:        </reason>
S:      </extValue>
S:    </result>
S:    <msgQ count="201" id="1">
S:      <qDate>2013-10-22T14:25:57.0Z</qDate>
S:      <msg>Registry initiated update of domain.</msg>
S:    </msgQ>
S:    <resData>
S:      <domain:infData
S:       xmlns:domain="urn:ietf:params:xml:ns:domain-1.0">
S:        <domain:name>domain.example</domain:name>
S:        <domain:roid>EXAMPLE1-REP</domain:roid>
S:        <domain:status s="ok"/>
S:        <domain:registrant>jd1234</domain:registrant>
S:        <domain:contact type="admin">sh8013</domain:contact>
S:        <domain:contact type="tech">sh8013</domain:contact>
S:        <domain:clID>ClientX</domain:clID>
S:        <domain:crID>ClientY</domain:crID>
S:        <domain:crDate>2012-04-03T22:00:00.0Z</domain:crDate>
S:        <domain:exDate>2014-04-03T22:00:00.0Z</domain:exDate>
S:      </domain:infData>
S:    </resData>
S:    <trID>
S:      <clTRID>ABC-12345</clTRID>
S:      <svTRID>54322-XYZ</svTRID>
S:    </trID>
S:  </response>
S:</epp>
```

Below is an unhandled domain name <info> <poll> response [RFC5731] and the unhandled <changePoll:changeData> element [RFC8590] included under an <extValue> element.

```
S:<?xml version="1.0" encoding="UTF-8" standalone="no"?>
S:<epp xmlns="urn:ietf:params:xml:ns:epp-1.0">
S:  <response>
S:    <result code="1301">
S:      <msg>Command completed successfully; ack to dequeue</msg>
S:      <extValue>
S:        <value>
S:          <domain:infData
S:            xmlns:domain="urn:ietf:params:xml:ns:domain-1.0">
S:            <domain:name>domain.example</domain:name>
S:            <domain:roid>EXAMPLE1-REP</domain:roid>
S:            <domain:status s="ok"/>
S:            <domain:registrant>jd1234</domain:registrant>
S:            <domain:contact type="admin">sh8013</domain:contact>
S:            <domain:contact type="tech">sh8013</domain:contact>
S:            <domain:clID>ClientX</domain:clID>
S:            <domain:crID>ClientY</domain:crID>
S:            <domain:crDate>2012-04-03T22:00:00.0Z</domain:crDate>
S:            <domain:exDate>2014-04-03T22:00:00.0Z</domain:exDate>
S:          </domain:infData>
S:        </value>
S:        <reason>
S:          urn:ietf:params:xml:ns:domain-1.0 not in login services
S:        </reason>
S:      </extValue>
S:      <extValue>
S:        <value>
S:          <changePoll:changeData
S:            xmlns:changePoll=
S:              "urn:ietf:params:xml:ns:changePoll-1.0"
S:            state="after">
S:            <changePoll:operation>update</changePoll:operation>
S:            <changePoll:date>
S:              2013-10-22T14:25:57.0Z</changePoll:date>
S:            <changePoll:svTRID>12345-XYZ</changePoll:svTRID>
S:            <changePoll:who>URS Admin</changePoll:who>
S:            <changePoll:caseId type="urs">urs123
S:            </changePoll:caseId>
S:            <changePoll:reason>URS Lock</changePoll:reason>
S:          </changePoll:changeData>
S:        </value>
S:        <reason>
S:        urn:ietf:params:xml:ns:changePoll-1.0 not in login services
S:        </reason>
S:      </extValue>
S:    </result>
S:    <msgQ count="201" id="1">
S:      <qDate>2013-10-22T14:25:57.0Z</qDate>
S:      <msg>Registry initiated update of domain.</msg>
S:    </msgQ>
S:    <trID>
S:      <clTRID>ABC-12345</clTRID>
S:      <svTRID>54322-XYZ</svTRID>
S:    </trID>
S:  </response>
S:</epp>
```

7. Implementation Considerations

There are implementation considerations for the client and the server to help address the risk of the client ignoring unhandled namespace information included in an EPP response that is needed to meet technical, policy, or legal requirements.

7.1. Client Implementation Considerations

To reduce the likelihood of a client receiving unhandled namespace information, the client should consider implementing the following:

1. Ensure that the client presents the complete set of what it supports when presenting its login services. If there are gaps between the services supported by the client and the login services included in the login command, the client may receive unhandled namespace information that the client could have supported.

2. Support all of the services included in the server greeting services that may be included in an EPP response, including the <poll> responses. The client should evaluate the gaps between the greeting services and the login services provided in the login command to identify extensions that need to be supported.

3. Proactively monitor for unhandled namespace information in the EPP responses by looking for the inclusion of the <extValue> element in successful responses, record the unsupported namespace included in the <reason> element, and record the unhandled namespace information included in the <value> element for later processing. The unhandled namespace should be implemented by the client to ensure that information is processed fully in future EPP responses.

7.2. Server Implementation Considerations

To assist the clients in recognizing unhandled namespaces, the server should consider implementing the following:

1. Monitor for returning unhandled namespace information to clients and report it to the clients out of band to EPP, so the clients can add support for the unhandled namespaces.

2. Look for the unhandled namespace support in the login services when returning optional unhandled namespace information in general EPP responses.

8. IANA Considerations

8.1. XML Namespace

This document uses URNs to describe XML namespaces conforming to a registry mechanism described in [RFC3688]. The following URI assignment has been made by IANA.

URI: urn:ietf:params:xml:ns:epp:unhandled-namespaces-1.0

Registrant Contact: IESG
XML: None. Namespace URIs do not represent an XML specification.

8.2. EPP Extension Registry

The EPP operational practice described in this document has been registered by IANA in the "Extensions for the Extensible Provisioning Protocol (EPP)" registry described in [RFC7451]. The details of the registration are as follows:

Name of Extension: "Extensible Provisioning Protocol (EPP) Unhandled Namespaces"
Document Status: Standards Track
Reference: RFC 9038
Registrant: IETF, <iesg@ietf.org>
TLDs: Any
IPR Disclosure: None
Status: Active
Notes: None

9. Security Considerations

This document does 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. Since the unhandled namespace content is XML that is not processed in the first pass by the XML parser, the client **SHOULD** validate the XML when the content is processed to protect against the inclusion of malicious content.

10. References

10.1. 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>.

[RFC5234]   Crocker, D., Ed. and P. Overell, "Augmented BNF for Syntax Specifications: ABNF", STD 68, RFC 5234, DOI 10.17487/RFC5234, January 2008, <https://www.rfc-editor.org/info/rfc5234>.

[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>.

[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>.

[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>.

10.2. Informative References

[RFC3735]   Hollenbeck, S., "Guidelines for Extending the Extensible Provisioning Protocol (EPP)", RFC 3735, DOI 10.17487/RFC3735, March 2004, <https://www.rfc-editor.org/info/rfc3735>.

[RFC3915]   Hollenbeck, S., "Domain Registry Grace Period Mapping for the Extensible Provisioning Protocol (EPP)", RFC 3915, DOI 10.17487/RFC3915, September 2004, <https://www.rfc-editor.org/info/rfc3915>.

[RFC5910]   Gould, J. and S. Hollenbeck, "Domain Name System (DNS) Security Extensions Mapping for the Extensible Provisioning Protocol (EPP)", RFC 5910, DOI 10.17487/RFC5910, May 2010, <https://www.rfc-editor.org/info/rfc5910>.

[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>.

[RFC8590]   Gould, J. and K. Feher, "Change Poll Extension for the Extensible Provisioning Protocol (EPP)", RFC 8590, DOI 10.17487/RFC8590, May 2019, <https://www.rfc-editor.org/info/rfc8590>.

Acknowledgements

Authors' Addresses

James Gould
VeriSign, Inc.
12061 Bluemont Way
Reston, VA 20190
United States of America
Email: jgould@verisign.com
URI: http://www.verisign.com

Martin Casanova
SWITCH
P.O. Box
CH-8021 Zurich
Switzerland
Email: martin.casanova@switch.ch
URI: http://www.switch.ch