Workgroup: Network Working Group
Internet-Draft: draft-ietf-regext-epp-eai-20
Published: 6 November 2023
Intended Status: Standards Track
Expires: 9 May 2024
Authors: D. Belyavskiy J. Gould
VeriSign, Inc.

Use of Internationalized Email Addresses in the Extensible Provisioning Protocol (EPP)

Abstract

This document describes an EPP command-response extension that permits usage of Internationalized Email Addresses in the EPP protocol and specifies the terms when it can be used by EPP clients and servers. The Extensible Provisioning Protocol (EPP), being developed before the standards for SMTPUTF8 compliant addresses, does not support such email addresses.

TO BE REMOVED on turning to RFC: The document is edited in the dedicated github repo. Please send your submissions via GitHub.

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 9 May 2024.

respect to this document. Code Components extracted from this document must include Revised BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Revised BSD License.

Table of Contents

1. Introduction

[RFC6530] introduced the framework for Internationalized Email Addresses. To make such addresses more widely accepted, the changes to various protocols need to be introduced.

This document describes an Extensible Provisioning Protocol (EPP) command-response extension, defined in [RFC5730], that permits the usage of Internationalized Email Addresses in the EPP protocol, specifies the terms when it can be used by EPP clients and servers, and defines an alternate email to use from ASCII-only to both ASCII and SMTPUTF8. The extension is used to apply the rules for the processing of email address elements in all of the [RFC5730] extensions negotiated in the EPP session, which include the object and command-responses extensions. The extension can be applied to any object or command-response extension that uses an email address.

The Extensible Provisioning Protocol (EPP) specified in [RFC5730] is a base document for object management operations and an extensible framework that maps protocol operations to objects. The specifics of various objects managed via EPP is described in separate documents. This document is only referring to an email address as a property of a managed object, such as the <contact:email> element in the EPP contact mapping [RFC5733] or the <org:email> element in the EPP organization mapping [RFC8543], and command-response extensions applied to a managed object.

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 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 the examples are provided only to illustrate element relationships and are not REQUIRED in the protocol.

The XML namespace prefix "eai" is used for the namespace "urn:ietf:params:xml:ns:epp:eai-1.0", 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. 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 URI with a higher version number than the prior versions.

3. Email Address Specification

Support of non-ASCII email address syntax is defined in RFC 6530 [RFC6530]. This mapping does not prescribe minimum or maximum lengths for character strings used to represent email addresses. The exact syntax of such addresses is described in Section 3.3 of [RFC6531]. The validation rules introduced in RFC 6531 MUST be followed when processing this extension.

The definition of email address in the EPP RFCs, including Section 2.6 of [RFC5733] and Section 4.1.2, 4.2.1, and 4.2.5 of [RFC8543], references [RFC5322] for the email address syntax. The XML schema definition in Section 4 of [RFC5733] and Section 5 of [RFC8543] defines the "email" element using the type "eppcom:minTokenType", which is defined in Section 4.2 of [RFC5730] as an XML schema "token" type with minimal length of one. The XML schema "token" type will fully support the use of SMTPUTF8 compliant addresses so the primary application of the extension is to apply the use of [RFC6531] instead of [RFC5322] for the email address syntax. Other

EPP extensions may follow the formal syntax definition using the XML schema type "eppcom:minTokenType" and the [RFC5322] format specification, where this extension applies to all EPP extensions with the same or similar definitions.

The email address format is formally defined in Section 3.4.1 of [RFC5322], which only consists of printable US-ASCII characters for both the local-part and the domain ABNF rules. [RFC6531] extends the Mailbox, Local-part and Domain ABNF rules in [RFC5321] to support "UTF8-non-ascii", defined in Section 3.1 of [RFC6532], for the local-part and U-label, defined in Section 2.3.2.1 of [RFC5890], for the domain. By applying the syntax rules of [RFC6531], the EPP extensions will change from supporting only ASCII characters to supporting Internationalized characters both in the email address local-part and domain-part.

4. SMTPUTF8 <eai:eai> Element

An alternate email can be set using the <eai:eai> element with the command and response extensions defined in Section 6. The <eai:eai> element contains the following child element:

<eai:email>: An element following the syntax in Section 3 for defining an alternate ASCII or SMTPUTF8 address. An empty <eai:email/> element unsets the alternate email in the Update Command (Section 6.2.5) and indicates the alternate email is not set in the Info Response (Section 6.1.2).

5. SMTPUTF8 Extension

5.1. Scope of Extension

The extension applies to all object extensions and command-response extensions negotiated in the EPP session that include email address properties. Examples include the <contact:email> element in the EPP contact mapping [RFC5733] or the <org:email> element in the EPP organization mapping [RFC8543]. All registry zones (e.g., top-level domains) authorized for the client in the EPP session apply. There is no concept of a per-client, per-zone, per-extension, or per-field setting that is used to indicate support for SMTPUTF8 compliant addresses, but instead it's a global setting that applies to the EPP session.

5.2. Signaling Client and Server Support

The client and the server can signal support for the extension using a namespace URI in the login and greeting extension services respectively. The namespace URI "urn:ietf:params:xml:ns:epp:eai-1.0" is used to signal support for the extension. The client includes the namespace URI in an <svcExtension> <extURI> element of the [RFC5730]

<login> Command. The server includes the namespace URI in an <svcExtension> <extURI> element of the [RFC5730] Greeting.

5.3. Extension Behavior

5.3.1. SMTPUTF8 compliant addresses Extension Negotiated

If both client and server have indicated the support of the SMTPUTF8 addresses during the session establishment, they MUST be able to process the SMTPUTF8 address in any message having an email property during the established EPP session. Below are the server and client obligations when the SMTPUTF8 extension has been successfuly negotiated in the EPP session.

The server MUST satisfy the following obligations when the SMTPUTF8 extension has been negotiated:

* Accept SMTPUTF8 compliant addresses for all email properties in the EPP session negotiated object extensions and command-response extensions. For example the <contact:email> element in [RFC5733] and the <org:email> element in [RFC8543].

* Accept SMTPUTF8 compliant addresses for all registry zones (e.g., top-level domains) authorized for the client in the EPP session.

* Email address validation based on SMTPUTF8 validation rules defined in Section 3

* Storage of email properties that support internationalized characters.

* Return SMTPUTF8 compliant addresses for all email properties in the EPP responses.

The client MUST satisfy the following obligations when the SMTPUTF8 extension has been negotiated:

* Provide SMTPUTF8 compliant addresses for all e-mail properties in the EPP session negotiated object extensions and command-response extensions. For example the <contact:email> element in [RFC5733] and the <org:email> element in [RFC8543].

* Provide SMTPUTF8 compliant addresses for all registry zones (e.g., top-level domains) authorized for the client in the EPP session.

* Accept SMTPUTF8 compliant addresses in the EPP responses for all email properties in the EPP session negotiated object extensions and command-response extensions.

5.3.2. SMTPUTF8 Extension Not Negotiated

The lack of SMTPUTF8 adresses support can cause data and functional issues, so an SMTPUTF8 supporting client or server needs to handle cases where the opposite party doesn't support SMTPUTF8 addresses processing. Below are the server and client obligations when the SMTPUTF8 extension is not negotiated due to the lack of support by the peer.

The SMTPUTF8 supporting server MUST satisfy the following obligations when the client does not support the SMTPUTF8 extension:

- When the email property is required in the EPP command, the server MUST validate the email property sent by the client using the ASCII email validation rules.

- When the email property is optional in the EPP command, if the client supplies the email property the server MUST validate the email property using the ASCII email validation rules.

- When the email property is required in the EPP response, the server MUST validate whether the email property is an SMTPUTF8 address and if so return the error code 2308 "Data management policy violation".

- When the email property is optional in the EPP response and is provided, the server MUST validate whether the email property is an SMTPUTF8 address and if so return the error code 2308 "Data management policy violation".

The SMTPUTF8 supporting client MUST satisfy the following obligations when the server does not support the SMTPUTF8 extension:

- When the email property is required in the EPP command and the email property is an SMTPUTF8 address, the client MUST provide an ASCII email address. The provided email address should provide a way to contact the registrant.

- When the email property is optional in the EPP command and the email property is an SMTPUTF8 address and client does not have an ASCII address providing a way to contact the registrant, the client MUST omit the email property. If the email property is provided, the client MUST provide an ASCII email address.

6. EPP Command Mapping

A detailed description of the EPP syntax and semantics can be found in the EPP core protocol specification [RFC5730]. This section defines the provisioning of an alternate email address.

6.1. EPP Query Commands

EPP provides three commands to retrieve object information: <check> to determine if an object can be provisioned, <info> to retrieve information associated with an object, and <transfer> to retrieve object-transfer status information.

6.1.1. EPP <check> Command

This extension does not add any elements to the EPP <check> command or <check> response described in [RFC5730].

6.1.2. EPP <info> Command

This extension does not add any elements to the EPP <info> command response described in [RFC5730].

If the query was successful, the server replies with an <eai:eai> element (Section 4) along with the regular EPP <resData>.

The following is an example <info> contact response using the <eai:eai> extension with no alternate email:

```
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:      <contact:infData
S:       xmlns:contact="urn:ietf:params:xml:ns:contact-1.0">
S:        <contact:id>sh8013</contact:id>
S:        <contact:roid>SH8013-REP</contact:roid>
S:        <contact:status s="linked"/>
S:        <contact:status s="clientDeleteProhibited"/>
S:        <contact:postalInfo type="int">
S:          <contact:name>John Doe</contact:name>
S:          <contact:org>Example Inc.</contact:org>
S:          <contact:addr>
S:            <contact:street>123 Example Dr.</contact:street>
S:            <contact:street>Suite 100</contact:street>
S:            <contact:city>Dulles</contact:city>
S:            <contact:sp>VA</contact:sp>
S:            <contact:pc>20166-6503</contact:pc>
S:            <contact:cc>US</contact:cc>
S:          </contact:addr>
S:        </contact:postalInfo>
S:        <contact:voice x="1234">+1.7035555555</contact:voice>
S:        <contact:fax>+1.7035555556</contact:fax>
S:        <contact:email>  @example.com</contact:email>
S:        <contact:clID>ClientY</contact:clID>
S:        <contact:crID>ClientX</contact:crID>
S:        <contact:crDate>1999-04-03T22:00:00.0Z</contact:crDate>
S:        <contact:upID>ClientX</contact:upID>
S:        <contact:upDate>1999-12-03T09:00:00.0Z</contact:upDate>
S:        <contact:trDate>2000-04-08T09:00:00.0Z</contact:trDate>
S:        <contact:authInfo>
S:          <contact:pw>2fooBAR</contact:pw>
S:        </contact:authInfo>
S:        <contact:disclose flag="0">
S:          <contact:voice/>
S:          <contact:email/>
S:        </contact:disclose>
S:      </contact:infData>
S:    </resData>
S:    <extension>
S:      <eai:eai
S:       xmlns:eai="urn:ietf:params:xml:ns:epp:eai-1.0">
S:        <eai:email/>
S:      </eai:eai>
S:    </extension>
```

```
S:    <trID>
S:      <clTRID>ABC-12345</clTRID>
S:      <svTRID>54322-XYZ</svTRID>
S:    </trID>
S:  </response>
S:</epp>
```

Figure 1: Example <info> contact response using the <eai:eai> extension with no alternate email

The following is an example <info> contact response using the <eai:eai> extension with an ASCII alternate email:

```
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:      <contact:infData
S:       xmlns:contact="urn:ietf:params:xml:ns:contact-1.0">
S:        <contact:id>sh8013</contact:id>
S:        <contact:roid>SH8013-REP</contact:roid>
S:        <contact:status s="linked"/>
S:        <contact:status s="clientDeleteProhibited"/>
S:        <contact:postalInfo type="int">
S:          <contact:name>John Doe</contact:name>
S:          <contact:org>Example Inc.</contact:org>
S:          <contact:addr>
S:            <contact:street>123 Example Dr.</contact:street>
S:            <contact:street>Suite 100</contact:street>
S:            <contact:city>Dulles</contact:city>
S:            <contact:sp>VA</contact:sp>
S:            <contact:pc>20166-6503</contact:pc>
S:            <contact:cc>US</contact:cc>
S:          </contact:addr>
S:        </contact:postalInfo>
S:        <contact:voice x="1234">+1.7035555555</contact:voice>
S:        <contact:fax>+1.7035555556</contact:fax>
S:        <contact:email>  @example.com</contact:email>
S:        <contact:clID>ClientY</contact:clID>
S:        <contact:crID>ClientX</contact:crID>
S:        <contact:crDate>1999-04-03T22:00:00.0Z</contact:crDate>
S:        <contact:upID>ClientX</contact:upID>
S:        <contact:upDate>1999-12-03T09:00:00.0Z</contact:upDate>
S:        <contact:trDate>2000-04-08T09:00:00.0Z</contact:trDate>
S:        <contact:authInfo>
S:          <contact:pw>2fooBAR</contact:pw>
S:        </contact:authInfo>
S:        <contact:disclose flag="0">
S:          <contact:voice/>
S:          <contact:email/>
S:        </contact:disclose>
S:      </contact:infData>
S:    </resData>
S:    <extension>
S:      <eai:eai
S:       xmlns:eai="urn:ietf:params:xml:ns:epp:eai-1.0">
S:        <eai:email>jdoe@example.com</eai:email>
S:      </eai:eai>
S:    </extension>
```

```
S:    <trID>
S:      <clTRID>ABC-12345</clTRID>
S:      <svTRID>54322-XYZ</svTRID>
S:    </trID>
S:  </response>
S:</epp>
```

Figure 2: Example <info> contact response using the <eai:eai> extension with an ASCII alternate email

The following is an example <info> contact response using the <eai:eai> extension with an SMTPUTF8 alternate email:

```
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:      <contact:infData
S:       xmlns:contact="urn:ietf:params:xml:ns:contact-1.0">
S:        <contact:id>sh8013</contact:id>
S:        <contact:roid>SH8013-REP</contact:roid>
S:        <contact:status s="linked"/>
S:        <contact:status s="clientDeleteProhibited"/>
S:        <contact:postalInfo type="int">
S:          <contact:name>John Doe</contact:name>
S:          <contact:org>Example Inc.</contact:org>
S:          <contact:addr>
S:            <contact:street>123 Example Dr.</contact:street>
S:            <contact:street>Suite 100</contact:street>
S:            <contact:city>Dulles</contact:city>
S:            <contact:sp>VA</contact:sp>
S:            <contact:pc>20166-6503</contact:pc>
S:            <contact:cc>US</contact:cc>
S:          </contact:addr>
S:        </contact:postalInfo>
S:        <contact:voice x="1234">+1.7035555555</contact:voice>
S:        <contact:fax>+1.7035555556</contact:fax>
S:        <contact:email>jdoe@example.com</contact:email>
S:        <contact:clID>ClientY</contact:clID>
S:        <contact:crID>ClientX</contact:crID>
S:        <contact:crDate>1999-04-03T22:00:00.0Z</contact:crDate>
S:        <contact:upID>ClientX</contact:upID>
S:        <contact:upDate>1999-12-03T09:00:00.0Z</contact:upDate>
S:        <contact:trDate>2000-04-08T09:00:00.0Z</contact:trDate>
S:        <contact:authInfo>
S:          <contact:pw>2fooBAR</contact:pw>
S:        </contact:authInfo>
S:        <contact:disclose flag="0">
S:          <contact:voice/>
S:          <contact:email/>
S:        </contact:disclose>
S:      </contact:infData>
S:    </resData>
S:    <extension>
S:      <eai:eai
S:       xmlns:eai="urn:ietf:params:xml:ns:epp:eai-1.0">
S:        <eai:email>  @example.com</eai:email>
S:      </eai:eai>
S:    </extension>
```

```
S:    <trID>
S:      <clTRID>ABC-12345</clTRID>
S:      <svTRID>54322-XYZ</svTRID>
S:    </trID>
S:  </response>
S:</epp>
```

Figure 3: Example <info> contact response using the <eai:eai> extension with an SMTPUTF8 alternate email

6.1.3. EPP <transfer> Query Command

This extension does not add any elements to the EPP <transfer> query command or <transfer> query response described in [RFC5730].

6.2. 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.

6.2.1. EPP <create> Command

This extension defines additional elements to extend the EPP <create> command of an object mapping like [RFC5733].

The EPP <create> command provides a transform operation that allows a client to create an instance of an object. In addition to the EPP command elements described in an object mapping like [RFC5733], the command MUST contain a child <eai:eai> element (Section 4) for the client to be set an alternate email. If the alternate email does not apply to the object, the server MUST return an EPP error result code of 2201.

The following is an example <create> command to create a contact object with an alternate ASCII email:

```
C:<?xml version="1.0" encoding="UTF-8" standalone="no"?>
C:<epp xmlns="urn:ietf:params:xml:ns:epp-1.0">
C:  <command>
C:    <create>
C:      <contact:create
C:       xmlns:contact="urn:ietf:params:xml:ns:contact-1.0">
C:        <contact:id>sh8013</contact:id>
C:        <contact: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:        </contact:postalInfo>
C:        <contact:voice x="1234">+1.7035555555</contact:voice>
C:        <contact:fax>+1.7035555556</contact:fax>
C:        <contact:email>  @example.com</contact:email>
C:        <contact:authInfo>
C:          <contact:pw>2fooBAR</contact:pw>
C:        </contact:authInfo>
C:        <contact:disclose flag="0">
C:          <contact:voice/>
C:          <contact:email/>
C:        </contact:disclose>
C:      </contact:create>
C:    </create>
C:    <extension>
C:      <eai:eai
C:       xmlns:eai="urn:ietf:params:xml:ns:epp:eai-1.0">
C:        <eai:email>jdoe@example.com</eai:email>
C:      </eai:eai>
C:    </extension>
C:    <clTRID>ABC-12345</clTRID>
C:  </command>
C:</epp>
```

Figure 4: Example <create> command to create a contact object with an alternate ASCII email

The following is an example <create> command to create a contact object with an alternate SMTPUTF8 email:

```
C:<?xml version="1.0" encoding="UTF-8" standalone="no"?>
C:<epp xmlns="urn:ietf:params:xml:ns:epp-1.0">
C:  <command>
C:    <create>
C:      <contact:create
C:       xmlns:contact="urn:ietf:params:xml:ns:contact-1.0">
C:        <contact:id>sh8013</contact:id>
C:        <contact: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:        </contact:postalInfo>
C:        <contact:voice x="1234">+1.7035555555</contact:voice>
C:        <contact:fax>+1.7035555556</contact:fax>
C:        <contact:email>jdoe@example.com</contact:email>
C:        <contact:authInfo>
C:          <contact:pw>2fooBAR</contact:pw>
C:        </contact:authInfo>
C:        <contact:disclose flag="0">
C:          <contact:voice/>
C:          <contact:email/>
C:        </contact:disclose>
C:      </contact:create>
C:    </create>
C:    <extension>
C:      <eai:eai
C:       xmlns:eai="urn:ietf:params:xml:ns:epp:eai-1.0">
C:        <eai:email>  @example.com</eai:email>
C:      </eai:eai>
C:    </extension>
C:    <clTRID>ABC-12345</clTRID>
C:  </command>
C:</epp>
```

Figure 5: Example <create> command to create a contact object with an alternate SMTPUTF8 email

This extension does not add any elements to the EPP <create> response described in [RFC5730].

6.2.2. EPP <delete> Command

This extension does not add any elements to the EPP <delete> command or <delete> response described in [RFC5730].

6.2.3. EPP <renew> Command

This extension does not add any elements to the EPP <renew> command or <renew> response described in [RFC5730].

6.2.4. EPP <transfer> Command

This extension does not add any elements to the EPP <transfer> command or <transfer> response described in [RFC5730].

6.2.5. EPP <update> Command

This extension defines additional elements to extend the EPP <update> command of an object mapping like [RFC5733].

The EPP <update> command provides a transform operation that allows a client to update an instance of an object. In addition to the EPP command elements described in an object mapping like [RFC5733], the command MUST contain a child <eai:eai> element (Section 4) for the client to be set or unset an alternate email. If the alternate email does not apply to the object, the server MUST return an EPP error result code of 2201.

The following is an example <update> command to set a contact object with an alternate ASCII email:

```
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:     <contact:update
C:      xmlns:contact="urn:ietf:params:xml:ns:contact-1.0">
C:       <contact:id>sh8013</contact:id>
C:     </contact:update>
C:   </update>
C:   <extension>
C:     <eai:eai
C:      xmlns:eai="urn:ietf:params:xml:ns:epp:eai-1.0">
C:       <eai:email>jdoe@example.com</eai:email>
C:     </eai:eai>
C:   </extension>
C:   <clTRID>ABC-12345</clTRID>
C: </command>
C:</epp>
```

Figure 6: Example <update> command to set a contact object with an alternate ASCII email

The following is an example <update> command to set a contact object with an alternate SMTPUTF8 email:

```
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:     <contact:update
C:      xmlns:contact="urn:ietf:params:xml:ns:contact-1.0">
C:       <contact:id>sh8013</contact:id>
C:     </contact:update>
C:   </update>
C:   <extension>
C:     <eai:eai
C:      xmlns:eai="urn:ietf:params:xml:ns:epp:eai-1.0">
C:       <eai:email>  @example.com</eai:email>
C:     </eai:eai>
C:   </extension>
C:   <clTRID>ABC-12345</clTRID>
C: </command>
C:</epp>
```

Figure 7: Example <update> command to set a contact object with an alternate SMTPUTF8 email

The following is an example <update> command to unset a contact object alternate email:

```
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:     <contact:update
C:      xmlns:contact="urn:ietf:params:xml:ns:contact-1.0">
C:       <contact:id>sh8013</contact:id>
C:     </contact:update>
C:   </update>
C:   <extension>
C:     <eai:eai
C:      xmlns:eai="urn:ietf:params:xml:ns:epp:eai-1.0">
C:       <eai:email/>
C:     </eai:eai>
C:   </extension>
C:   <clTRID>ABC-12345</clTRID>
C: </command>
C:</epp>
```

Figure 8: Example <update> command to unset a contact object alternate email

This extension does not add any elements to the EPP <create> response described in [RFC5730].

7. Formal Syntax

The EPP SMTPUTF8 Addresses Extension schema is presented here.

The formal syntax shown here is a complete XML 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 XML Schema; they are used to note the beginning and ending of the XML Schema for URI registration purposes.

7.1. SMTPUTF8 Addresses Schema

```
<CODE BEGINS>
<?xml version="1.0" encoding="UTF-8"?>
<schema xmlns="http://www.w3.org/2001/XMLSchema"
  xmlns:eai="urn:ietf:params:xml:ns:epp:eai-1.0"
  targetNamespace="urn:ietf:params:xml:ns:epp:eai-1.0"
  elementFormDefault="qualified">
  <annotation>
    <documentation>Extensible Provisioning Protocol v1.0
       SMTPUTF8 Addresses Schema.</documentation>
  </annotation>
  <!-- Create, Update, and Info Response extension element -->
  <element name="eai" type="eai:eaiType" />
  <!--
    Single email element that can be empty
   -->
  <complexType name="eaiType">
    <sequence>
      <element name="email"
        type="token"/>
    </sequence>
  </complexType>
  <!--
 End of schema.
 -->
</schema>
<CODE ENDS>
```

8. IANA Considerations

8.1. XML Namespace

This document uses URNs to describe XML namespaces conforming to a registry mechanism described in RFC 3688 [RFC3688]. The following URI assignment should be made by IANA:

Registration request for the eai namespace:

URI: urn:ietf:params:xml:ns:epp:eai-1.0
Registrant Contact: IESG
XML: None. Namespace URIs do not represent an XML specification.

Registration request for the eai XML Schema:

URI: urn:ietf:params:xml:schema:epp:eai-1.0
Registrant Contact: IESG
XML: See the "Formal Syntax" section of this document.

8.2. EPP Extension Registry

The EPP extension described in this document should be registered by IANA in the "Extensions for the Extensible Provisioning Protocol (EPP)" registry described in RFC 7451 [RFC7451]. The details of the registration are as follows:

Name of Extension: "Use of Internationalized Email Addresses in EPP protocol"
Document status: Standards Track
Reference: RFC 8807
Registrant Name and Email Address: IESG, <iesg@ietf.org>
Top-Level Domains(TLDs): Any
IPR Disclosure: None
Status: Active
Notes: None

9. Implementation Status

Note to RFC Editor: Please remove this section and the reference to RFC 7942 [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 RFC 7942 [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 RFC 7942 [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".

9.1. Verisign EPP SDK

Organization: Verisign Inc.

Name: Verisign EPP SDK

Description: The Verisign EPP SDK includes both a full client implementation and a full server stub implementation of draft-ietf-regext-epp-eai.

Level of maturity: Development

Coverage: All aspects of the protocol are implemented.

Licensing: GNU Lesser General Public License

Contact: jgould@verisign.com

URL: https://www.verisign.com/en_US/channel-resources/domain-registry-products/epp-sdks

10. Security Considerations

As is noted in Section 10.1 and Section 13 of [RFC6530], unconstrained Unicode in email addresses can introduce a class of security threats that do not exist with all-ASCII email addresses. As EPP exists in ecosystems where email addresses passed in EPP are displayed in RDAP and other services and copy-and-paste of these email addresses is common for businesses transferring domains via EPP, there should be safeguards against these threats. Therefore, usage of the SMTPUTF8 email addresses as specified in this document SHOULD be done with policies that disallow the use of unconstrained Unicode. The domain-part of these SMTPUTF8 email addresses SHOULD conform to IDNA2008. The local-part of these SMTPUTF8 email addresses SHOULD be restricted to Unicode that does not introduce the threats noted in [RFC6530]. One such possible solution would be to disallow characters outside of Unicode Annex 31 (https://unicode.org/reports/tr31/).

As email address is often a primary end user contact, an invalid email address may put the communication with the end user into risk in case when such contact is necessary. In case of an invalid domain name in the email address a malicious actor can register a valid domain name with similar U-label (homograph attack) and get a control over the domain name associated with the contact using social engineering techniques. To reduce the risk of the use of invalid domain names in email addresses, registries **SHOULD** validate the domain name syntax in the provided email addresses and validate whether the domain name consists of the code points allowed by IDNA Rules and Derived Property Values.

When the SMTPUTF8 extension is negotiated by both the client and the server, the client and server obligations defined in Section 5.3.1 MUST be satisfied. If the obligations are not satisfied by either the client or server, the SMTPUTF8 address may be mishandled in processing or storage and be unusable.

11. Acknowledgments

The authors would like to thank Alexander Mayrhofer, Chris Lonvick, Gustavo Lozano, Jody Kolker, John C Klensin, John Levine, Klaus Malorny, Marc Blanchet, Marco Schrieck, Mario Loffredo, Murray S. Kucherawy, Patrick Mevzek, Pete Resnick, Scott Hollenbeck, Takahiro Nemoto, Taras Heichenko, and Thomas Corte for their careful review and valuable comments.

12. References

12.1. Normative References

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.27487/RFC2119, March 1997, <https://www.rfc-editor.org/info/rfc2119>.

[RFC3688] Mealling, M., "The IETF XML Registry", BCP 81, RFC 3688, DOI 10.27487/RFC3688, January 2004, <https://www.rfc-editor.org/info/rfc3688>.

[RFC5321] Klensin, J., "Simple Mail Transfer Protocol", RFC 5321, DOI 10.17487/RFC5321, October 2008, <https://www.rfc-editor.org/info/rfc5321>.

[RFC5322] Resnick, P., Ed., "Internet Message Format", RFC 5322, DOI 10.17487/RFC5322, October 2008, <https://www.rfc-editor.org/info/rfc5322>.

[RFC5730] Hollenbeck, S., "Extensible Provisioning Protocol (EPP)", STD 69, RFC 5730, DOI 10.27487/RFC5730, August 2009, <https://www.rfc-editor.org/info/rfc5730>.

[RFC5733] Hollenbeck, S., "Extensible Provisioning Protocol (EPP) Contact Mapping", STD 69, RFC 5733, DOI 10.27487/RFC5733, August 2009, <https://www.rfc-editor.org/info/rfc5733>.

[RFC5890] Klensin, J., "Internationalized Domain Names for Applications (IDNA): Definitions and Document Framework", RFC 5890, DOI 10.17487/RFC5890, August 2010, <https://www.rfc-editor.org/info/rfc5890>.

[RFC6530] Klensin, J. and Y. Ko, "Overview and Framework for Internationalized Email", RFC 6530, DOI 10.17487/RFC6530, February 2012, <https://www.rfc-editor.org/info/rfc6530>.

[RFC6531] Yao, J. and W. Mao, "SMTP Extension for Internationalized Email", RFC 6531, DOI 10.17487/RFC6531, February 2012, <https://www.rfc-editor.org/info/rfc6531>.

[RFC6532] Yang, A., Steele, S., and N. Freed, "Internationalized Email Headers", RFC 6532, DOI 10.17487/RFC6532, February 2012, <https://www.rfc-editor.org/info/rfc6532>.

[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>.

[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>.

[RFC8174] Leiba, B., "Ambiguity of Uppercase vs Lowercase in RFC 2119 Key Words", BCP 14, RFC 8174, DOI 10.27487/RFC8174, May 2017, <https://www.rfc-editor.org/info/rfc8174>.

12.2. Informative References

[RFC7451] Hollenbeck, S., "Extension Registry for the Extensible Provisioning Protocol", RFC 7451, DOI 10.27487/RFC7451, February 2015, <https://www.rfc-editor.org/info/rfc7451>.

[RFC8543] Zhou, L., Kong, N., Yao, J., Gould, J., and G. Zhou, "Extensible Provisioning Protocol (EPP) Organization Mapping", RFC 8543, DOI 10.27487/RFC8543, March 2019, <https://www.rfc-editor.org/info/rfc8543>.

Appendix A. Change History

A.1. Change from 00 to 01

1. Changed from update of RFC 5733 to use the "Placeholder Text and a New Email Element" EPP Extension approach.

A.2. Change from 01 to 02

1. Fixed the XML schema and the XML examples based on validating them.

2. Added James Gould as co-author.

3. Updated the language to apply to any EPP object mapping and to use the EPP contact mapping as an example.

4. Updated the structure of document to be consistent with the other Command-Response Extensions.

5. Replaced the use of "eppEAI" in the XML namespace and the XML namespace prefix with "eai".

6. Changed to use a pointed XML namespace with "0.2" instead of "1.0".

A.3. Change from 02 to 03

1. The approach has changed to use the concept of Functional EPP Extension.

2. The examples are removed

A.4. Change from 03 to 04

1. More detailed reference to email syntax is provided

2. The shortened eai namespace reference is removed

A.5. Change from 04 to the regext 01 version

1. Provided the recommended placeholder value

A.6. Change from the regext 01 to regext 02 version

1. Removed the concept of the placeholder value

A.7. Change from the regext 02 to regext 03 version

1. Changed to use a pointed XML namespace with "0.3" instead of "0.2".

2. Some wording improvements

A.8. Change from the regext 03 to regext 04 version

1. Some nitpicking

A.9. Change from the regext 04 to regext 05 version

1. Some nitpicking

2. The "Implementation considerations" section is removed

A.10. Change from the regext 05 to regext 06 version

1. Some nitpicking

A.11. Change from the regext 06 to regext 07 version

1. Namespace version set to 1.0

A.12. Change from the regext 07 to regext 08 version

1. Information about implementations is provided.

2. Acknowledgments section is added.

3. Reference to RFC 7451 is moved to Informative.

4. IPR information is provided

5. Sections are reordered to align with the other regext documents

A.13. Change from the regext 08 to regext 09 version

1. Nitpicking according to Murray S. Kucherawy review

A.14. Change from the regext 09 to regext 10 version

1. Some nitpicking in the security considerations.

A.15. Change from the regext 10 to regext 11 version

1. Nitpicking according mostly GenArt review.

A.16. Change from the regext 11 to regext 12 version

1. XML schema registration request removed.

A.17. Change from the regext 12 to regext 13 version

1. Document updated according to SecDir and ART-ART review.

A.18. Change from the regext 13 to regext 14 version

1. Document updated according the IANA review #1231866.

A.19. Change from the regext 14 to regext 15 version

1. Document updated according to ART-ART review.

A.20. Change from the regext 15 to regext 16 version

1. Document removed the definition of the concept of a functional extension and updated to use a command-response extension, based on the feedback from John C Klensin.

2. Document removed the EAI abbreviation and uses SMTPUTF8 as umbrella term instead, based on the feedback from John C Klensin.

A.21. Change from the regext 16 to regext 17 version

1. Added support for an alternate email during a transition period, based on feedback from John C Klensin.

A.22. Change from the regext 17 to regext 18 version

1. Roll back to approach in -16 with the Cardinality of One Option, posted to and supported on the mailing list.

2. Replaced references of eai to smtputf8, based on feedback from John C Klensin.

3. Revised the Security Considerations section based on feedback and text from Andy Newton.

A.23. Change from the regext 18 to regext 19 version

1. Reverted back to -17 with support for one or two email addresses using either ASCII or SMTPUTF8 and remove any reference to the requirement for an ASCII email address and remove the concept of a transition period.

A.24. Change from the regext 19 to regext 20 version

1. Reverted Security Considerations section back to the content in -18 based on feedback from Andy Newton.

Authors' Addresses

Dmitry Belyavskiy
8 marta st.
Moscow
127083
Russian Federation

Phone: +7 916 262 5593
Email: beldmit@gmail.com

James Gould
VeriSign, Inc.
12061 Bluemont Way
Reston, VA 20190
United States of America

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