Network Working Group J. Gould
Internet-Draft VeriSign, Inc.
Intended status: Standards Track K. Feher
Expires: April 7, 2019 Neustar
October 04, 2018

Allocation Token Extension for the Extensible Provisioning Protocol (EPP)
draft-ietf-regext-allocation-token-12

Abstract

This document describes an Extensible Provisioning Protocol (EPP) extension for including an Allocation Token in "query" and "transform" commands. The Allocation Token is used as a credential that authorizes a client to request the allocation of a specific object from the server, using one of the EPP transform commands including create and transfer.

Status of This Memo

Copyright Notice

Table of Contents

1. Introduction

This document describes an extension mapping for version 1.0 of the Extensible Provisioning Protocol (EPP) [RFC5730]. This mapping, an extension to EPP object mappings like the EPP domain name mapping [RFC5731], supports passing an Allocation Token as a credential that authorizes a client to request the allocation of a specific object from the server, using one of the EPP transform commands including create and transfer.

Allocation is when a server assigns the sponsoring client of an object based on the use of an Allocation Token credential. Examples include allocating a registration based on a pre-eligibility Allocation Token, allocating a premium domain name registration based on an auction Allocation Token, allocating a registration based on a founders Allocation Token, and allocating an existing domain name held by the server or by a different sponsoring client based on an Allocation Token passed with a transfer command.

Clients pass an Allocation Token to the server for validation, and the server determines if the supplied Allocation Token is one supported by the server. It is up to server policy which EPP transform commands and which objects require the Allocation Token. The Allocation Token MAY be returned to an authorized client for passing out-of-band to a client that uses it with an EPP transform command.

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 "allocationToken" is used for the namespace "urn:ietf:params:xml:ns:allocationToken-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.

The "abc123" token value is used as a placeholder value in the examples. The server MUST support token values that follow the Security Considerations (Section 7) section.

The domain object attribute values, including the "2fooBAR" <domain:pw> value, in the examples are provided for illustration purposes only. Refer to [RFC5731] for details on the domain object attributes.

2. Object Attributes

This extension adds additional elements to EPP object mappings like the EPP domain name mapping [RFC5731]. Only those new elements are described here.

2.1. Allocation Token

The Allocation Token is a simple XML "token" type. The exact format of the Allocation Token is up to server policy. The server MAY have the Allocation Token for each object to match against the Allocation Token passed by the client to authorize the allocation of the object. The <allocationToken:allocationToken> element is used for all of the supported EPP commands as well as the info response. If the supplied Allocation Token passed to the server does not apply to the object, the server MUST return an EPP error result code of 2201.

Authorization information, like what is defined in the EPP domain name mapping [RFC5731], is associated with objects to facilitate transfer operations. The authorization information is assigned when an object is created. The Allocation Token and the authorization information are both credentials, but used for different purposes and used in different ways. The Allocation Token is used to facilitate the allocation of an object instead of transferring the sponsorship of the object. The Allocation Token is not managed by the client, but is validated by the server to authorize assigning the initial sponsoring client of the object.

An example <allocationToken:allocationToken> element with value of "abc123":

```
<allocationToken:allocationToken xmlns:allocationToken=
          "urn:ietf:params:xml:ns:allocationToken-1.0">
  abc123
</allocationToken:allocationToken>
```

3. EPP Command Mapping

A detailed description of the EPP syntax and semantics can be found in the EPP core protocol specification [RFC5730].

3.1. 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.

3.1.1. EPP <check> Command

This extension defines additional elements to extend the EPP <check> command of an object mapping like [RFC5731].

This extension allows clients to check the availability of an object with an Allocation Token, as described in Section 2.1. Clients can check if an object can be created using the Allocation Token. The Allocation Token is applied to all object names included in the EPP <check> command.

Example <check> command for the allocation.example domain name using the <allocationToken:allocationToken> extension with the allocation token of 'abc123':

```
C:<?xml version="1.0" encoding="UTF-8" standalone="no"?>
C:<epp xmlns="urn:ietf:params:xml:ns:epp-1.0">
C:  <command>
C:    <check>
C:      <domain:check
C:       xmlns:domain="urn:ietf:params:xml:ns:domain-1.0">
C:        <domain:name>allocation.example</domain:name>
C:      </domain:check>
C:    </check>
C:    <extension>
C:      <allocationToken:allocationToken
C:        xmlns:allocationToken=
C:          "urn:ietf:params:xml:ns:allocationToken-1.0">
C:        abc123
C:      </allocationToken:allocationToken>
C:    </extension>
C:    <clTRID>ABC-12345</clTRID>
C:  </command>
C:</epp>
```

If the query was successful, the server replies with a <check> response providing the availability status of the queried object based on the following Allocation Token cases, where the object is otherwise available:

1. If an object requires an Allocation Token and the Allocation Token does apply to the object, then the server MUST return the availability status as available (e.g., "avail" attribute is "1" or "true").
2. If an object requires an Allocation Token and the Allocation Token does not apply to the object, then the server SHOULD return the availability status as unavailable (e.g., "avail" attribute is "0" or "false").
3. If an object does not require an Allocation Token, the server MAY return the availability status as available (e.g., "avail" attribute is "1" or "true").

Example <check> domain response for a <check> command using the <allocationToken:allocationToken> extension:

```
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 lang="en-US">Command completed successfully</msg>
S:  </result>
S:  <resData>
S:   <domain:chkData
S:     xmlns:domain="urn:ietf:params:xml:ns:domain-1.0">
S:    <domain:cd>
S:     <domain:name avail="1">allocation.example</domain:name>
S:    </domain:cd>
S:   </domain:chkData>
S:  </resData>
S:  <trID>
S:   <clTRID>ABC-DEF-12345</clTRID>
S:   <svTRID>54321-XYZ</svTRID>
S:  </trID>
S: </response>
S:</epp>
```

Example <check> command with the <allocationToken:allocationToken> extension for the allocation.example and allocation2.example domain names. Availability of allocation.example and allocation2.example domain names are based on the Allocation Token 'abc123':

```
C:<?xml version="1.0" encoding="UTF-8"?>
C:<epp xmlns="urn:ietf:params:xml:ns:epp-1.0">
C: <command>
C:  <check>
C:   <domain:check
C:     xmlns:domain="urn:ietf:params:xml:ns:domain-1.0">
C:    <domain:name>allocation.example</domain:name>
C:    <domain:name>allocation2.example</domain:name>
C:   </domain:check>
C:  </check>
C:  <extension>
C:   <allocationToken:allocationToken
C:     xmlns:allocationToken=
C:       "urn:ietf:params:xml:ns:allocationToken-1.0">
C:     abc123
C:   </allocationToken:allocationToken>
C:  </extension>
C:  <clTRID>ABC-DEF-12345</clTRID>
C: </command>
C:</epp>
```

Example <check> domain response for multiple domain names in the <check> command using the <allocationToken:allocationToken> extension, where the Allocation Token 'abc123' matches allocation.example but does not match allocation2.example:

```
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 lang="en-US">Command completed successfully</msg>
S:  </result>
S:  <resData>
S:   <domain:chkData
S:     xmlns:domain="urn:ietf:params:xml:ns:domain-1.0">
S:    <domain:cd>
S:     <domain:name avail="1">allocation.example</domain:name>
S:    </domain:cd>
S:    <domain:cd>
S:     <domain:name avail="0">allocation2.example</domain:name>
S:     <domain:reason>Allocation Token mismatch</domain:reason>
S:    </domain:cd>
S:   </domain:chkData>
S:  </resData>
S:  <trID>
S:   <clTRID>ABC-DEF-12345</clTRID>
S:   <svTRID>54321-XYZ</svTRID>
S:  </trID>
S: </response>
S:</epp>
```

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

3.1.2. EPP <info> Command

This extension defines additional elements to extend the EPP <info> command of an object mapping like [RFC5731].

The EPP <info> command allows a client to request information associated with an existing object. Authorized clients MAY retrieve the Allocation Token (Section 2.1) along with the other object information by supplying the <allocationToken:info> element in the command. The <allocationToken:info> element is an empty element that serves as a marker to the server to return the <allocationToken:allocationToken> element in the info response. If the client is not authorized to receive the Allocation Token, the server MUST return an EPP error result code of 2201. If the client

is authorized to receive the Allocation Token, but there is no Allocation Token associated with the object, the server MUST return an EPP error result code of 2303. The authorization is subject to server policy.

Example <info> command with the allocationToken:info extension for the allocation.example domain name:

```
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:    <domain:info
C:      xmlns:domain="urn:ietf:params:xml:ns:domain-1.0"
C:      xsi:schemaLocation="urn:ietf:params:xml:ns:domain-1.0
C:      domain-1.0.xsd">
C:      <domain:name>allocation.example</domain:name>
C:    </domain:info>
C:   </info>
C:   <extension>
C:      <allocationToken:info
C:        xmlns:allocationToken=
C:          "urn:ietf:params:xml:ns:allocationToken-1.0/>
C:   </extension>
C:   <clTRID>ABC-12345</clTRID>
C:  </command>
C:</epp>
```

If the query was successful, the server replies with an <allocationToken:allocationToken> element along with the regular EPP <resData>. The <allocationToken:allocationToken> element is described in Section 2.1.

Example <info> domain response using the <allocationToken:allocationToken> extension:

```
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:      <domain:infData
S:       xmlns:domain="urn:ietf:params:xml:ns:domain-1.0">
S:        <domain:name>allocation.example</domain:name>
S:        <domain:roid>EXAMPLE1-REP</domain:roid>
S:        <domain:status s="pendingCreate"/>
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:authInfo>
S:          <domain:pw>2fooBAR</domain:pw>
S:        </domain:authInfo>
S:      </domain:infData>
S:    </resData>
S:    <extension>
S:      <allocationToken:allocationToken
S:        xmlns:allocationToken=
S:          "urn:ietf:params:xml:ns:allocationToken-1.0">
S:        abc123
S:      </allocationToken:allocationToken>
S:    </extension>
S:    <trID>
S:      <clTRID>ABC-12345</clTRID>
S:      <svTRID>54321-XYZ</svTRID>
S:    </trID>
S:  </response>
S:</epp>
```

3.1.3. EPP <transfer> Query Command

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

3.2. EPP Transform Commands

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

3.2.1. EPP <create> Command

This extension defines additional elements to extend the EPP <create> command of an object mapping like [RFC5731].

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 [RFC5731], the command MUST contain a child <allocationToken:allocationToken> element for the client to be authorized to create and allocate the object. If the Allocation Token does not apply to the object, the server MUST return an EPP error result code of 2201.

Example <create> command to create a domain object with an Allocation Token:

```
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:      <domain:create
C:       xmlns:domain="urn:ietf:params:xml:ns:domain-1.0">
C:        <domain:name>allocation.example</domain:name>
C:        <domain:registrant>jd1234</domain:registrant>
C:        <domain:contact type="admin">sh8013</domain:contact>
C:        <domain:contact type="tech">sh8013</domain:contact>
C:        <domain:authInfo>
C:          <domain:pw>2fooBAR</domain:pw>
C:        </domain:authInfo>
C:      </domain:create>
C:    </create>
C:    <extension>
C:      <allocationToken:allocationToken
C:        xmlns:allocationToken=
C:          "urn:ietf:params:xml:ns:allocationToken-1.0">
C:        abc123
C:      </allocationToken:allocationToken>
C:    </extension>
C:    <clTRID>ABC-12345</clTRID>
C:  </command>
C:</epp>
```

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

3.2.2. EPP <delete> Command

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

3.2.3. EPP <renew> Command

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

3.2.4. EPP <transfer> Command

This extension defines additional elements to extend the EPP <transfer> request command of an object mapping like [RFC5731].

The EPP <transfer> request command provides a transform operation that allows a client to request the transfer of an object. In addition to the EPP command elements described in an object mapping like [RFC5731], the command MUST contain a child <allocationToken:allocationToken> element for the client to be authorized to transfer and allocate the object. The authorization associated with the Allocation Token is in addition to and does not replace the authorization mechanism defined for the object's <transfer> request command. If the Allocation Token is invalid or not required for the object, the server MUST return an EPP error result code of 2201.

Example <transfer> request command to allocate the domain object with the Allocation Token:

```
C:<?xml version="1.0" encoding="UTF-8" standalone="no"?>
C:<epp xmlns="urn:ietf:params:xml:ns:epp-1.0">
C:  <command>
C:    <transfer op="request">
C:      <domain:transfer
C:        xmlns:domain="urn:ietf:params:xml:ns:domain-1.0">
C:        <domain:name>example1.tld</domain:name>
C:        <domain:period unit="y">1</domain:period>
C:        <domain:authInfo>
C:          <domain:pw>2fooBAR</domain:pw>
C:        </domain:authInfo>
C:      </domain:transfer>
C:    </transfer>
C:    <extension>
C:      <allocationToken:allocationToken
C:        xmlns:allocationToken=
C:          "urn:ietf:params:xml:ns:allocationToken-1.0">
C:        abc123
C:      </allocationToken:allocationToken>
C:    </extension>
C:    <clTRID>ABC-12345</clTRID>
C:  </command>
C:</epp>
```

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

3.2.5. EPP <update> Command

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

4. Formal Syntax

One schema is presented here that is the EPP Allocation Token Extension schema.

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

4.1. Allocation Token Extension Schema

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

  <!-- Element used in info command to get allocation token. -->
  <element name="info">
    <complexType>
      <complexContent>
        <restriction base="anyType" />
      </complexContent>
    </complexType>
  </element>

  <!-- Allocation Token used in transform
    commands and info response -->
  <element name="allocationToken"
    type="allocationToken:allocationTokenType" />
  <simpleType name="allocationTokenType">
    <restriction base="token">
      <minLength value="1" />
    </restriction>
  </simpleType>

<!-- End of schema. -->
</schema>
END
```

5. IANA Considerations

5.1. XML Namespace

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

Registration request for the allocationToken namespace:

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

Registration request for the allocationToken XML schema:

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

5.2. EPP Extension Registry

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

Name of Extension: "Allocation Token Extension for the Extensible Provisioning Protocol (EPP)"

Document status: Standards Track

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

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

TLDs: Any

IPR Disclosure: None

Status: Active

Notes: None

6. 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".

6.1. 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-allocation-token.

Level of maturity: Production

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

6.2. Neustar EPP SDK

Organisation: Neustar Inc.

Name: Neustar EPP SDK

Description: The Neustar EPP SDK includes a full client implementation of draft-ietf-regext-allocation-token.

Level of maturity: Production

Coverage: All aspects of the protocol are implemented.

Licensing: GNU Lesser General Public License

Contact: quoc-anh.np@team.neustar

URL: http://registrytoolkit.neustar

6.3. Neustar gTLD SRS

Organisation: Neustar Inc.

Name: Neustar generic Top Level Domain (gTLD) Shared Registry System (SRS).

Description: The Neustar gTLD SRS implements the server side of draft-ietf-regext-allocation-token for several Top Level Domains.

Level of maturity: Production

Coverage: All server side aspects of the protocol are implemented.

Licensing: Proprietary

Contact: quoc-anh.np@team.neustar

6.4. Net::DRI

Organization: Dot and Co

Name: Net::DRI

Description: Net::DRI implements the client-side of draft-ietf-regext-allocation-token.

Level of maturity: Production

Coverage: All client-side aspects of the protocol are implemented.

Licensing: GNU Lesser General Public License

Contact: netdri@dotandco.com

7. Security Considerations

The mapping described in 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.

The mapping acts as a conduit for the passing of Allocation Tokens between a client and a server. The definition of the Allocation Token SHOULD be defined outside of this mapping. The following are security considerations in the definition and use of an Allocation Token:

1. An Allocation Token should be considered secret information by the client and SHOULD be protected at rest and MUST be protected in transit.
2. An Allocation Token should be single use, meaning it should be unique per object and per allocation operation.
3. An Allocation Token should have a limited life with some form of expiry in the Allocation Token if generated by a trusted 3rd third party, or with a server-side expiry if generated by the server.
4. An Allocation Token should use a strong random value if it is based on an unsigned code.
5. An Allocation Token should leverage digital signatures to confirm its authenticity if generated by a trusted 3rd party.
6. An Allocation Token that is signed XML should be encoded (e.g., base64 [RFC4648]) to mitigate server validation issues.

8. Acknowledgements

The authors wish to acknowledge the original concept for this draft and the efforts in the initial versions of this draft by Trung Tran and Sharon Wodjenski.

Special suggestions that have been incorporated into this document were provided by Ben Campbell, Scott Hollenbeck, Benjamin Kaduk, Mirja Kuehlewind, Rubens Kuhl, Alexander Mayrhofer, Patrick Mevzek, Eric Rescoria, and Adam Roach.

9. References

9.1. Normative References

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

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

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

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

[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>.

9.2. Informative References

[RFC4648] Josefsson, S., "The Base16, Base32, and Base64 Data Encodings", RFC 4648, DOI 10.17487/RFC4648, October 2006, <https://www.rfc-editor.org/info/rfc4648>.

[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>.

[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>.

Appendix A. Change History

A.1. Change from 00 to 01

1. Amended XML Namespace section of IANA Considerations, added EPP Extension Registry section.
2. Moved Change History to the back section as an Appendix.

A.2. Change from 01 to 02

1. Ping update.

A.3. Change from 02 to 03

1. Ping update.

A.4. Change from 03 to 04

1. Updated the authors for the draft.

A.5. Change from 04 to REGEXT 00

1. Changed to regext working group draft by changing draft-gould-allocation-token to draft-ietf-regext-allocation-token.

A.6. Change from REGEXT 00 to REGEXT 01

1. Ping update.

A.7. Change from REGEXT 01 to REGEXT 02

1. Added the Implementation Status section.

A.8. Change from REGEXT 02 to REGEXT 03

1. Changed Neustar author to Kal Feher.

A.9. Change from REGEXT 03 to REGEXT 04

1. Added Neustar implementation to the Implementation Status section.

A.10. Change from REGEXT 04 to REGEXT 05

1. Updates based on feedback from Patrick Mevzek, that include:

    1. Remove "or code" from the Abstract section.
    2. Add a missing "to" in "an allocation token TO one of the EPP..." in the Introduction section.
    3. Reword the "The allocation token is known to the server..." sentence in the Introduction section.
    4. Modify the "The allocation token MAY be returned to an authorized client for passing out-of-band to a client that uses it with an EPP transform command" to clarify who the two separate clients are.
    5. Removed an unneeded ":" from the EPP <transfer> Command and EPP <update> Command sections.

A.11. Change from REGEXT 05 to REGEXT 06

1. Fix description of Neustar gTLD SRS based on feedback from Rubens Kuhl.
2. Updates based on feedback from Alexander Mayrhofer, that include:

    1. Making all references to Allocation Token to use the upper case form.
    2. Revise the language of the abstract to include "for including an Allocation Token in query and transform commands. The Allocation Token is used as a credential that authorizes a client to request the allocation of a specific object from the server, using one of the EPP transform commands..."
    3. Replace the title "EPP <transfer> Command" with "EPP <transfer> Query Command" for section 3.1.3.
    4. Revise the second sentence of the Introduction to "The mapping, ..., supports passing an Allocation Token..."
    5. Change "support" to "require" in the Introduction sentence "It is up to server policy which EPP transform commands and which objects support the Allocation Token."
    6. Add the definition of Allocation to the Introduction.
    7. Removed "transform" from "all of the supported EPP transform commands" in the "Allocation Token" section, since the Allocation Token can be used with the "check" command as well.
    8. Remove the word "same" from "The same <allocationToken:allocationToken> element is used for all..." in the "Allocation Token" section.
    9. Change the description of the use of the 2201 error in the "Allocation Token" section, the "EPP <create> Command" section, the "EPP <transfer> Command" section, and the "EPP <update> Command" section.
    10. Revise "<check> to determine if an object is known to the server..." to "<check> to determine if an object can be provisioned..." and remove "detailed" in the description of the <info> in the "EPP Query Commands" section.
    11. Add missing description of the expected <check> response behavior.
    12. Replaced the example reason "Invalid domain-token pair" with "Allocation Token mismatch".
    13. Replace "information on" with "information associated with" in the "EPP <info> Command" section.
    14. Removed the "that identifies the extension namespace", the ", defined in...", the Allocation Token links from the error response sentences, and the "object referencing the <allocationToken:info> element" in the "EPP <info> Command" section.

    15. Added "The authorization is subject to server policy." to the "EPP <info> Command" section.
    16. Replace "or <transfer> response>" with "or <transfer> query response>" in the EPP <transfer> Query Command" section.
    17. Replace "create an object" with "create an instance of an object" in the "EPP <create> Command" section.
    18. Revised the sentence to include "the command MUST contain a child <allocationToken:allocationToken> element for the client to be authorized to create and allocate the object" in the "EPP <create> Command" section.
    19. Removed the reference to section 2.1 and the namespace identification text in the "EPP <transfer> Command" section.
    20. Added "The authorization associated with the Allocation Token is in addition to and does not replace the authorization mechanism defined for the object's <transfer> request command." to the "EPP <transfer> Command" section.
    21. Modified the first sentence of the "EPP Extension Registry" section to read "The following registration of the EPP Extension Registry, described in RFC7451, is requested"
    22. Removed support with using the Allocation Token with an empty extension of update (e.g., release command), based on the confusion and lack of known applicability.

3. Updates based on feedback from Scott Hollenbeck, that include:

    1. Revised XML schema to included a minimum length of 1 for the allocationTokenType.
    2. Revised the "IANA Considerations" section to include the registration of the XML schema.
    3. Revised the "Security Considerations" section to include considerations for the definition of the Allocation Tokens.

A.12. Change from REGEXT 06 to REGEXT 07

1. Updates based on feedback from Patrick Mevzek:

    1. Updated obsoleted RFC 7942 to RFC 7942.
    2. Moved RFC 7451 to an informational reference.

A.13. Change from REGEXT 07 to REGEXT 08

1. Changed Kal Feher's contact e-mail address.
2. Changed Neustar's Implementation Status contact e-mail address.
3. Added the Net::DRI sub-section to the Implementation Status section.

A.14. Change from REGEXT 08 to REGEXT 09

1. Updates based on the AD review by Adam Roach, that include:
    1. In "Abstract", set "query" and "transform" off in some way (e.g., using quotation marks)
    2. In "Conventions Used in This Document", please update to use the boilerplate from RFC 8174.
    3. Remove "allocationToken-1.0" is used as an abbreviation for "urn:ietf:params:xml:ns:allocationToken-1.0".
    4. In "Allocation Token", change "The server MUST have the Allocation Token" to "The server MAY have the Allocation Token".
    5. In "EPP <check> Command", change "This extension allow clients" to "This extension allows clients".
    6. Use domains reserved by RFC 2026 for the examples. The example domain "example.tld" was changed to "allocation.example" and the example domain "example2.tld" was changed to "allocation2.example".
    7. In "EPP <info> Command", change "...the server MUST return an EPP error result code of 2303 object referencing the <allocationToken:info> element." to "...the server MUST return an EPP error result code of 2303."
    8. In "EPP <transfer> Query Command", remove "the" before "RFC5730".
    9. In "EPP <transfer> Command", change "If the Allocation Token does not apply to the object..." to "If the Allocation Token is invalid or not required for the object...".
    10. In "XML Namespace", remove the sentence "The following URI assignment is requested of IANA:"
    11. In "Security Considerations", change "An Allocation Token should is" to "An Allocation Token that is". Also informatively cite RFC 4648 for the base64 reference.
2. Change "ietf:params:xml:ns:allocationToken-1.0" to "ietf:params:xml:schema:allocationToken-1.0" for the XML schema IANA registration.

A.15. Change from REGEXT 09 to REGEXT 10

1. Changed "auhorization" to "authorization" in the "EPP <info> Command" section.
2. Added 'If an object does not require an Allocation Token, the server MAY return the availability status as available (e.g., "avail" attribute is "1" or "true").' to the check response cases, based on feedback by Mirja Kuehlewind.
3. Changed the definition of the <info> element in the XML schema to only allow an empty element, based on IANA's expert review.

4. Added normative language to the storage and transport of the Allocation Token, in the "Security Considerations" section, based on feedback from Eric Rescoria.
5. Changed "The definition of the Allocation Token is defined outside of this mapping" to "The definition of the Allocation Token SHOULD be defined outside of this mapping", in the "Security Considerations" section, based on feedback from Eric Rescoria.
6. Added the missing "urn:" prefix with the IANA URI registrations.
7. The URL for the BCP 14 was removed based on feedback from Alissa Cooper.
8. Updates based on review by Benjamin Kaduk, that include:

    1. Added the second paragraph to the "Allocation Token" section to describe the difference (motivation) of using the Allocation Token versus the EPP RFC authorization mechanism.
    2. Added a paragraph to the "Conventions Used in This Document" section for the use of the "abc123" token value and the use of domain object "2fooBAR" password value in the examples.
    3. Changed the "A client MUST pass an Allocation Token known to the server to be authorized to use one of the supported EPP transform commands." sentence in the "Introduction" section to "Clients pass an Allocation Token to the server for validation, and the server determines if the supplied Allocation Token is one supported by the server."
    4. Changed the "Indentation and white space in the examples are provided only to illustrate element relationships and are not REQUIRED in the protocol." sentence in the "Conventions Used in This Document" section to "Indentation and white space in the examples are provided only to illustrate element relationships and are not REQUIRED in the protocol."
    5. Changed the "Authorized clients MAY retrieve..." sentence in the "EPP <info> Command" section.
    6. Changed the "If the query was successful..." sentence in the "EPP <info> Command" section.
    7. Added "supplied" to the "If the supplied Allocation Token passed..." sentence in the "Allocation Token" section.
    8. Removed an extra newline in the <annotation> element in the "Allocation Token Extension Schema" section.

A.16. Change from REGEXT 10 to REGEXT 11

1. Removed the old duplicate "Authorized clients MAY retrieve..." sentence from section 3.1.2 "EPP <info> Command".

A.17. Change from REGEXT 11 to REGEXT 12

1. Revised the example <check> domain response to first include the positive case for allocation.example, and to second include the negative case for allocation2.example, based on feedback from Ben Campbell. The caption was revised for the example to include the text ", where the Allocation Token 'abc123' matches allocation.example but does not match allocation2.example".

Authors' Addresses

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

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

Kal Feher
Neustar
lvl 8/10 Queens Road
Melbourne, VIC 3004
AU

Email: ietf@feherfamily.org
URI: http://www.neustar.biz