   Launch Phase Mapping for the Extensible Provisioning Protocol (EPP)
                     draft-tan-epp-launchphase-01

Abstract

   This document describes an Extensible Provisioning Protocol (EPP)
   extension mapping for the provisioning and management of domain names
   during the launch phase of a domain name registry.

Status of this Memo

   This Internet-Draft is submitted in full conformance with the
   provisions of BCP 78 and BCP 79.

   Internet-Drafts are working documents of the Internet Engineering
   Task Force (IETF). Note that other groups may also distribute
   working documents as Internet-Drafts. The list of current Internet-
   Drafts is at http://datatracker.ietf.org/drafts/current/.

   Internet-Drafts are draft documents valid for a maximum of six months
   and may be updated, replaced, or obsoleted by other documents at any
   time. It is inappropriate to use Internet-Drafts as reference
   material or to cite them other than as "work in progress."

   This Internet-Draft will expire on April 16, 2012.

Copyright Notice

Table of Contents

1. Introduction

   This document describes an extension mapping for version 1.0 of the
   Extensible Provisioning Protocol (EPP) [RFC5730]. This EPP mapping
   specifies a flexible schema that can be used to implement several
   common use cases related to the provisioning and management of launch
   phase extension in a domain name registry.

   It is typical for domain registries to operate in special modes
   within certain periods of time to facilitate allocation of domain
   names for a subset of the zone namespace that becomes available.
   This document uses the term "launch phase" to refer to such a period.

   The EPP domain name mapping [RFC5731] is designed for the steady
   state operation of a registry. During a launch phase, however,
   registries typically accept multiple applications for a given domain
   name. This document proposes an extension to the domain name
   extension in order to unambiguously manage the received applications.

1.1. Conventions Used in This Document

   The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT",
   "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this
   document are to be interpreted as described in RFC 2119 [RFC2119].

   XML is case sensitive. Unless stated otherwise, XML specifications
   and examples provided in this document MUST be interpreted in the
   character case presented in order to develop a conforming
   implementation.

   "launchphase-1.0" is used as an abbreviation for
   "urn:ietf:params:xml:ns:launchphase-1.0". The XML namespace prefix
   "lp" is used, 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. Application Object

   It is common for domain registries to allow multiple applications of
   a given domain name during its launch phase operations.

   Upon receiving a request to create a domain, the server creates an
   application object corresponding to the request and assigns an
   identifier for the application and returns it to the client. This
   mapping defines an "applicationID" element for this purpose.

   In order to facilitate correlation, all subsequent operations on the

   domain object MUST be qualified by the previously assigned
   applicationID.

   To support common use cases of launch phase operations, this mapping
   also defines several other elements that may be used in
   implementations.

2.1. <lp:phase> Element

   To allow for multiple simultaneous launch phases, the application
   object MAY also include an <lp:phase> element whose content is a
   server-defined opaque identifier corresponding to each launch phase.
   Depending on the policy of the domain registry, the phase may be
   implicit (based on the time of request or encoded as part of the
   applicationID) or explicitly required.

2.2. <lp:status> Element

   The <status> element is used to convey extended status(es) pertaining
   to the application object, beyond what is specified in the object
   mapping to which this application object represents.

   The following status values are defined:

   pending: the initial state of a newly-created application object.

   validated: the application meets relevant registry rules.

   invalid: the application does not validate according to registry
      rules

   allocated: one of two possible end states of an application object;
      the object corresponding to the application has been provisioned

   rejected: the other possible end state; the object was not
      provisioned

   Certain status values MAY be combined. For example, an application
   can be invalid and rejected. [[Q1: Should we allow multiples? --WT]]

2.2.1. State Transition

```
                             | request
                             v
                      +---------+
                      | pending |
                      +----+----+
                           |
                           |
        +--------------+-----+----------+--------------+
        |              |          |          |
        v              v          v          v
   +-----------+   +---------+   +-------+   +-------+
   |           |   |         |   /       \   /       \
   | validated |   | invalid +----->| rejected  |   | allocated |
   |           |   |         |   \       /   \       /
   +----+------+   +----+----+   +-------+   +-------+
        |              |             ^           ^
        |              |             |           |
        |              +---------------+         |
        |              |             |           |
        |              |             |           |
        +------------------------------+         |
        |              |             |           |
        +-----------------------------------------+
```

                             Figure 1
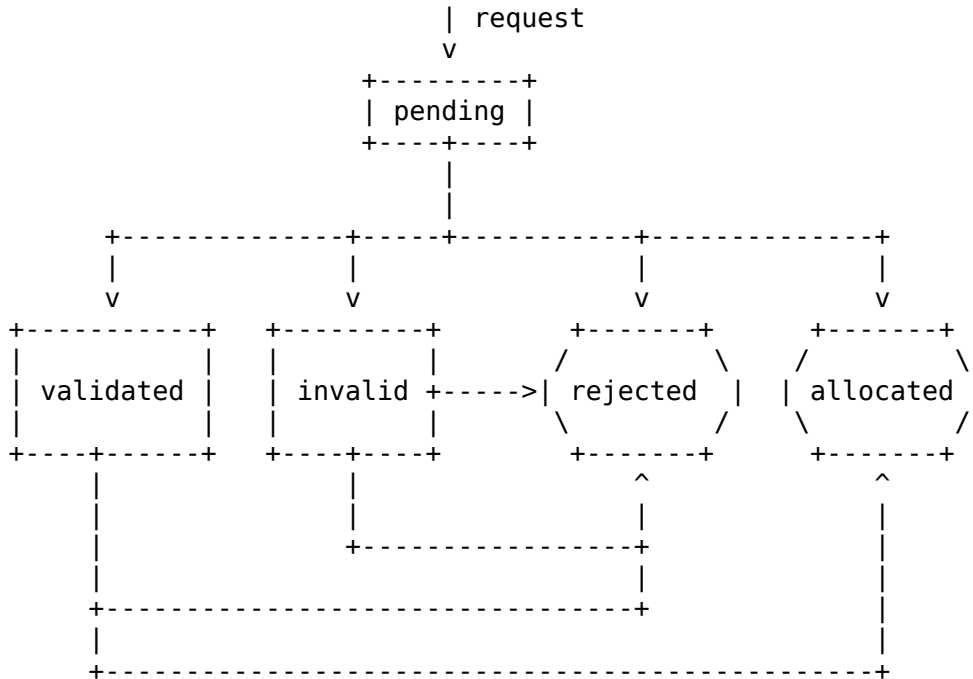
2.3. Claim Elements

   An application may have one or more <claim> elements. A <claim>
   element describes an applicant's prior right to a given domain name.

   The <claim> element has the boolean "preValidated" attribute, which
   indicates whether a third party validation agency has already
   validated the claim. When this attribute has a true value, the
   <pvrc> element must always be present.

   Several child elements of the <claim> element are defined:

   <pvrc>: The Pre-Validation Result Code, an opaque string issued by a
      third-party validation agent

   <claimIssuer>: contains the ID of a contact object (as described in
      RFC 5733 [RFC5733]) identifying the contact information of the
      authority which issued the right (for example, a trade mark office
      or company registration bureau)

   <claimName>: identifying the text string in which the applicant is
      claiming a prior right

   <claimNumber>: the registration number of the right (ie trademark
      number or company registration number)

   <claimType>: indicates the type of claim being made (eg trademark,
      symbol, combined mark, company name)

   <claimEntitlement>: indicates the applicant's entitlement to the
      claim (ie, owner or licensee)

   <claimRegDate>: the date of registration of the claim

   <claimExDate>: the date of expiration of the claim

   <claimCountry>: indicates the country in which the claim is valid

   <claimRegion>: indicates the name of a city, state, province or
      other geographic region in which the claim is valid. This may be
      a two-character code from [WIPO.ST3]

3. EPP Command Mapping

   This mapping is designed to be flexible, requiring only a minimum set
   of required elements.

   While it is meant to serve several use cases, it does not prescribe
   any interpretation by the client or server. Such processing is
   typically highly policy-dependent and therefore specific to
   implementations.

   Operations on application objects are done via one or more of the
   existing EPP verbs defined in the EPP domain mapping. Registries may
   choose to support a subset of the operations.

3.1. EPP <check> Command

   This extension does not define any extension to the EPP <check>
   command or response described in the EPP domain name mapping
   [RFC5731].

3.2. EPP <info> Command

   This extension defines additional elements to extend the EPP <info>
   command and response to be used in conjunction with the domain name
   mapping.

In order to indicate that the query is meant for an application
object, an <lp:info> element is sent along with the regular <info>
domain command. The <lp:info> element contains the following child
elements:

<lp:applicationID> the application identifier for which the client
        wishes to query.

<lp:phase> (optional) the phase during which the application was
        submitted or is associated with.

If the query was successful, the server replies with an <lp:infData>
element along with the regular EPP <resData>. The <lp:infData>
contains the following child elements:

<lp:applicationID> the application identifier of the returned
        application.

<lp:phase> (optional) the phase during which the application was
        submitted or is associated with.

<lp:status> (optional) status of the application.

<lp:claim> (optional) one or more <lp:claim> elements.

If present, the <lp:claim> elements may contain the following child
elements:

<pvrc>: The Pre-Validation Result Code.

<claimIssuer>: the ID of a contact object representing the issuing
        authority.

<claimName>: the textual representation of the right.

<claimNumber>: the registration number.

<claimType>: the type of claim being made.

<claimEntitlement>: the entitlement.

<claimRegDate>: the registration date.

<claimExDate>: the expiry date.

   <claimCountry>: the country.

   <claimRegion>: the geographic region.

3.2.1. Client Processing Considerations

   The client MUST ensure that any successful <info> command results in
   a response that an <lp:infData> element is returned in the response.
   This serves as a cross check that the server did receive the query
   for the application (and not a domain of the same name) and processed
   it as it was intended.

3.2.2. Example <info> command

   Following is an example <info> domain command with the <lp:info>
   extension.

```
<?xml version="1.0" encoding="UTF-8" standalone="no"?>
<epp xmlns="urn:ietf:params:xml:ns:epp-1.0">
  <command>
    <info>
      <domain:info
          xmlns:domain="urn:ietf:params:xml:ns:domain-1.0">
        <domain:name>example.tld</domain:name>
      </domain:info>
    </info>
    <extension>
      <lp:info xmlns:lp="urn:ietf:params:xml:ns:launchphase-1.0">
        <lp:applicationID>2393-9323-E08C-03B1</lp:applicationID>
        <lp:phase>phase1</lp:phase>
      </lp:info>
    </extension>
    <clTRID>example:epp:239331</clTRID>
  </command>
</epp>
```

   An example response that corresponds to the above command.

```
<?xml version="1.0" encoding="utf-8"?>
<epp xmlns="urn:ietf:params:xml:ns:epp-1.0">
  <response>
    <result code="1000">
      <msg>Command completed successfully</msg>
    </result>
    <resData>
      <domain:infData xmlns:domain="urn:ietf:params:xml:ns:domain-1.0">
        <domain:name>example.tld</domain:name>
        <domain:roid>32302393_TESTDOMAIN-TLD</domain:roid>
```

```
        <domain:status s="pendingCreate" />
        <domain:registrant>ga3000</domain:registrant>
        <domain:contact type="admin">ue312987</domain:contact>
        <domain:contact type="tech">ue312987</domain:contact>
        <domain:contact type="billing">ue312987</domain:contact>
        <domain:ns>
          <domain:hostObj>ns1.example.com</domain:hostObj>
          <domain:hostObj>ns2.example.net</domain:hostObj>
        </domain:ns>
        <domain:clID>client1</domain:clID>
        <domain:crID>client1</domain:crID>
        <domain:crDate>2010-09-18T06:12:39.0Z</domain:crDate>
        <domain:authInfo>
          <domain:pw>foo!bar#baz</domain:pw>
        </domain:authInfo>
      </domain:infData>
    </resData>
    <extension>
      <lp:infData xmlns:lp="urn:ietf:params:xml:ns:launchphase-1.0">
        <lp:applicationID>2393-9323-E08C-03B1</lp:applicationID>
        <lp:phase>phase1</lp:phase>
        <lp:status s="pending" />
        <lp:claim>
          <lp:pvrc>3828590-P1F-932391651E3A2900338C12</lp:pvrc>
          <lp:claimIssuer>CONTACT-IPCLEARINGHOUSE</lp:claimIssuer>
          <lp:claimName>Hello</lp:claimName>
          <lp:claimNumber>GE 3933232</lp:claimNumber>
          <lp:claimType>REG-TM-WORD</lp:claimType>
          <lp:claimEntitlement>owner</lp:claimEntitlement>
          <lp:claimRegDate>2011-09-09</lp:claimRegDate>
          <lp:claimExDate>2013-09-09</lp:claimExDate>
          <lp:claimCountry>AU</lp:claimCountry>
          <lp:claimCountry>VIC</lp:claimCountry>
        </lp:claim>
      </lp:infData>
    </extension>
    <trID>
      <clTRID>example:epp:239331</clTRID>
      <svTRID>server-8551292e23a</svTRID>
    </trID>
  </response>
 </epp>
```

3.3. EPP <create> Command

   This extension defines additional elements to extend the EPP <create>
   command and response to be used in conjunction with the domain name
   mapping.

   The EPP <create> command is used to create an application. Typically
   additional information is required to submit a domain name
   application during a launch phase. This extension introduces an <lp:
   create> to encapsulate commonly used fields. Another use case that
   extension addresses is the plausible need for a registry to
   distinguish between multiple (possibly concurrent) launch phases.
   Clients may specify the <lp:phase> in which the application is meant
   to be submitted. The <lp:create> element contains the following
   child elements.

   <lp:phase> (optional) the phase during which the application was
        submitted or is associated with.

   <lp:claim> (optional) one or more <lp:claim> elements.

   The format of the <lp:claim> element is identical to that specified
   in the section on EPP <info> command.

   Upon successful processing, the server assigns an application
   identifier and returns it in an <lp:creData> element together with
   the regular <resData>. The <lp:creData> element contains a single
   <lp:applicationID> element as described below:

   <lp:applicationID> the application identifier assigned by the
        server.

3.3.1. Example <create> command

   Following is an example <create> domain command with the <lp:create>
   extension.

```
<?xml version="1.0" encoding="UTF-8" standalone="no"?>
<epp xmlns="urn:ietf:params:xml:ns:epp-1.0">
  <command>
    <create>
      <domain:create
          xmlns:domain="urn:ietf:params:xml:ns:domain-1.0">
        <domain:name>example.tld</domain:name>
        <domain:period unit="y">2</domain:period>
        <domain:ns>
          <domain:hostObj>ns1.example.com</domain:hostObj>
          <domain:hostObj>ns2.example.net</domain:hostObj>
        </domain:ns>
        <domain:registrant>ga3000</domain:registrant>
        <domain:contact type="admin">ue312987</domain:contact>
        <domain:contact type="tech">ue312987</domain:contact>
        <domain:contact type="billing">ue312987</domain:contact>
        <domain:authInfo>
          <domain:pw>foo!bar#baz</domain:pw>
        </domain:authInfo>
      </domain:create>
    </create>
    <extension>
      <lp:create xmlns:lp="urn:ietf:params:xml:ns:launchphase-1.0">
        <lp:phase>phase1</lp:phase>
        <lp:claim>
          <lp:pvrc>3828590-P1F-932391651E3A2900338C12</lp:pvrc>
          <lp:claimIssuer>CONTACT-IPCLEARINGHOUSE</lp:claimIssuer>
          <lp:claimName>Hello</lp:claimName>
          <lp:claimNumber>GE 3933232</lp:claimNumber>
          <lp:claimType>REG-TM-WORD</lp:claimType>
          <lp:claimEntitlement>owner</lp:claimEntitlement>
          <lp:claimRegDate>2011-09-09</lp:claimRegDate>
          <lp:claimExDate>2013-09-09</lp:claimExDate>
          <lp:claimCountry>AU</lp:claimCountry>
          <lp:claimCountry>VIC</lp:claimCountry>
        </lp:claim>
      </lp:create>
    </extension>
    <clTRID>example:epp:239332</clTRID>
  </command>
</epp>
```

An example response that corresponds to the above command.

```
<?xml version="1.0" encoding="UTF-8" standalone="no"?>
<epp xmlns="urn:ietf:params:xml:ns:epp-1.0">
  <response>
    <result code="1000">
      <msg>Command completed successfully</msg>
    </result>
    <resData>
      <domain:creData
          xmlns:domain="urn:ietf:params:xml:ns:domain-1.0">
       <domain:name>example.tld</domain:name>
       <domain:crDate>2010-08-10T15:38:26.623854Z</domain:crDate>
       <domain:exDate>2012-08-10T15:38:26.623854Z</domain:exDate>
      </domain:creData>
    </resData>
    <extension>
      <lp:creData xmlns:lp="urn:ietf:params:xml:ns:launchphase-1.0">
        <lp:applicationID>2393-9323-E08C-03B1</lp:applicationID>
      </lp:creData>
    </extension>
    <trID>
      <clTRID>example:epp:239332</clTRID>
      <svTRID>server-8551292e23b</svTRID>
    </trID>
  </response>
</epp>
```

3.3.2. Client Processing Considerations

   The client MUST ensure that any successful <info> command results in
   a response that an <lp:infData> element is returned in the response.
   This serves as a cross check that the server did receive the query
   for the application (and not a domain of the same name) and processed
   it as it was intended.

3.4. EPP <update> Command

   This extension defines additional elements to extend the EPP <update>
   command to be used in conjunction with the domain name mapping.

   Registry policies permitting, clients may update an application
   object by submitting an EPP <update> command along with an <lp:
   update> element to indicate the application object to be updated.
   The <lp:update> element contains the following child elements:

   <lp:applicationID> the application identifier for which the client
         wishes to update.

   <lp:phase> (optional) the phase during which the application was
         submitted or is associated with.

   This extension does not define any extension to the response of an
   <update> domain command. After processing the command, the server
   replies with a standard EPP response as defined in the EPP domain
   mapping.

3.4.1. Server Processing Considerations

   A server implementation that conforms to this specification MUST
   respect and process the <lp:update> section, if present, and MUST
   respond with an error if the applicationID does not correspond with
   the domain name in the <domain:name> element.

3.4.2. Example <update> command

   Following is an example <update> domain command with the <lp:update>
   extension.

```
<?xml version="1.0" encoding="UTF-8" standalone="no"?>
<epp xmlns="urn:ietf:params:xml:ns:epp-1.0">
  <command>
    <update>
      <domain:update
          xmlns:domain="urn:ietf:params:xml:ns:domain-1.0">
        <domain:name>example.tld</domain:name>
        <domain:add>
          <domain:ns>
            <domain:hostObj>ns3.example.org</domain:hostObj>
          </domain:ns>
        </domain:add>
        <domain:rem>
          <domain:ns>
            <domain:hostObj>ns2.example.net</domain:hostObj>
          </domain:ns>
        </domain:rem>
        <domain:chg>
          <domain:registrant>n3o2999</domain:registrant>
        </domain:chg>
      </domain:update>
    </update>
    <extension>
      <lp:update xmlns:lp="urn:ietf:params:xml:ns:launchphase-1.0">
        <lp:applicationID>2393-9323-E08C-03B1</lp:applicationID>
        <lp:phase>phase1</lp:phase>
      </lp:update>
    </extension>
    <clTRID>example:epp:239333</clTRID>
  </command>
</epp>
```

   An example response that corresponds to the above command.

```
<?xml version="1.0" encoding="UTF-8"?>
<epp xmlns="urn:ietf:params:xml:ns:epp-1.0">
  <response>
    <result code="1000">
      <msg>Command completed successfully</msg>
    </result>
    <trID>
      <clTRID>example:epp:239333</clTRID>
      <svTRID>server-8551292e23c</svTRID>
    </trID>
  </response>
</epp>
```

3.5. EPP <delete> Command

   This extension defines additional elements to extend the EPP <delete>
   command to be used in conjunction with the domain name mapping.

   Registry policies permitting, clients may withdraw an application by
   submitting an EPP <delete> command along with an <lp:delete> element
   to indicate the application object to be deleted. The <lp:delete>
   element contains the following child elements:

   <lp:applicationID> the application identifier for which the client
          wishes to delete.

   <lp:phase> (optional) the phase during which the application was
          submitted or is associated with.

   This extension does not define any extension to the response of an
   <delete> domain command. After processing the command, the server
   replies with a standard EPP response as defined in the EPP domain
   mapping.

3.5.1. Server Processing Considerations

   A server implementation that conforms to this specification MUST
   respect and process the <lp:delete> section, if present, and MUST
   respond with an error if the applicationID does not correspond with
   the domain name in the <domain:name> element.

   Depending on the server policy, an implementation may choose to
   delete the application object immediately if business rules allow.
   In that case, the server MUST respond with an EPP 1000 result code.
   Alternatively, the server may choose to cancel the application
   object, in which case it SHOULD respond with an EPP 1001 result code

   to indicate that the object will be purged at a later date.

3.5.2. Example <delete> command

   Following is an example <delete> domain command with the <lp:delete>
   extension.

   <?xml version="1.0" encoding="UTF-8" standalone="no"?>
   <epp xmlns="urn:ietf:params:xml:ns:epp-1.0">
     <command>
       <delete>
         <domain:delete
             xmlns:domain="urn:ietf:params:xml:ns:domain-1.0">
           <domain:name>example.tld</domain:name>
         </domain:delete>
       </delete>
       <extension>
         <lp:delete xmlns:lp="urn:ietf:params:xml:ns:launchphase-1.0">
           <lp:applicationID>2393-9323-E08C-03B1</lp:applicationID>
           <lp:phase>phase1</lp:phase>
         </lp:delete>
       </extension>
       <clTRID>example:epp:239334</clTRID>
     </command>
   </epp>

   An example response that corresponds to the above command.

   <?xml version="1.0" encoding="UTF-8"?>
   <epp xmlns="urn:ietf:params:xml:ns:epp-1.0">
     <response>
       <result code="1000">
         <msg>Command completed successfully</msg>
       </result>
       <trID>
         <clTRID>example:epp:239334</clTRID>
         <svTRID>server-8551292e23d</svTRID>
       </trID>
     </response>
   </epp>

3.6. EPP <renew> Command

   This extension does not define any extension to the EPP <renew>
   command or response described in the EPP domain name mapping
   [RFC5731].

3.7. EPP <transfer> Command

   This extension does not define any extension to the EPP <transfer>
   command or response described in the EPP domain name mapping
   [RFC5731].

4. Formal Syntax

   [TBD]

5. Acknowledgements

   [to be filled in]

6. IANA Considerations

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

   Registration request for the extension namespace:

   URI: urn:ietf:params:xml:ns:launchphase-1.0

   Registrant Contact: IESG

   XML: None. Namespace URIs do not represent an XML specification.

   Registration request for the extension XML schema:

   URI: urn:ietf:params:xml:schema:launchphase-1.0

7. Security Considerations

   The mapping extensions described in this document do not provide any
   security services beyond those described by EPP [RFC5730], the EPP
   domain name mapping [RFC5731], and protocol layers used by EPP. The
   security considerations described in these other specifications apply
   to this specification as well.

   Updates to, and deletion of an application object must be restricted
   to clients authorized to perform the said operation on the object.

   As information contained within an application, or even the mere fact
   that an application exists may be confidential. Any attempt to

operate on an application object by an unauthorized client MUST be
rejected with an EPP 2303 (object does not exist) or an appropriate
auhorization error. Server policy may allow <info> operation with
filtered output by clients other than the sponsoring client, in which
case the <domain:infData> and <lp:infData> response SHOULD be
filtered to include only fields that are publicly accessible.

8. Normative References

   [RFC2119] Bradner, S., "Key words for use in RFCs to Indicate
              Requirement Levels", BCP 14, RFC 2119, March 1997.

   [RFC3688] Mealling, M., "The IETF XML Registry", BCP 81, RFC 3688,
              January 2004.

   [RFC5730] Hollenbeck, S., "Extensible Provisioning Protocol (EPP)",
              STD 69, RFC 5730, August 2009.

   [RFC5731] Hollenbeck, S., "Extensible Provisioning Protocol (EPP)
              Domain Name Mapping", STD 69, RFC 5731, August 2009.

   [RFC5733] Hollenbeck, S., "Extensible Provisioning Protocol (EPP)
              Contact Mapping", STD 69, RFC 5733, August 2009.

Authors' Addresses

   Wil Tan
   Cloud Registry
   Suite 32 Seabridge House
   377 Kent St
   Sydney, NSW 2000
   AU

   Phone: +61 414 710899
   Email: wil@cloudregistry.net
   URI: http://www.cloudregistry.net

Gavin Brown
CentralNic Ltd
35-39 Mooregate
London, England EC2R 6AR
GB

Phone: +44 8700 170 900
Email: gavin.brown@centralnic.com
URI: http://www.centralnic.com