Location-to-Service Translation (LoST) Service List Boundary Extension

Abstract

   Location-to-Service Translation (LoST) maps service identifiers and
   location information to service contact URIs. If a LoST client wants
   to discover available services for a particular location, it will
   perform a <listServicesByLocation> query to the LoST server.
   However, the LoST server, in its response, does not provide context
   information; that is, it does not provide any additional information
   about the geographical region within which the returned list of
   services is considered valid. Therefore, this document defines a
   Service List Boundary that returns a local context along with the
   list of services returned, in order to assist the client in not
   missing a change in available services when moving.

Status of This Memo

Table of Contents

1. Introduction

   Since the LoST protocol [RFC5222] employs the Service Boundary
   concept in order to avoid having clients continuously trying to
   refresh the mapping of a specific service, a Service List Boundary
   mechanism provides similar advantages for Service Lists.

   Location-based service providers, as well as Public Safety Answering
   Points (PSAPs), only serve a specific geographic region. Therefore,
   the LoST protocol defines the Service Boundary, which indicates the
   service region for a specific service URL. However, not all services
   are available everywhere. Clients can discover available services
   for a particular location via the <listServicesByLocation> query in
   LoST. The LoST server returns a list of services that are available
   at this particular location, but the server does not provide any
   additional information about the geographical region within which the
   returned Service List is considered valid. This may lead to the
   situation where a client initially discovers all available services
   via the <listServicesByLocation> query, and then moves to a different
   location (while refreshing the service mappings), but without
   noticing the availability of other services. The following imaginary
   example illustrates the problem for emergency calling:

   The client is powered-up, does location determination (resulting in
   location A), and performs an initial <listServicesByLocation> query
   with location A requesting urn:services:sos.

   The LoST server returns the following list of services:

      urn:service:sos.police
      urn:service:sos.ambulance
      urn:service:sos.fire

   The client does the initial LoST mapping and discovers the
   dialstrings for each service. Then the client moves, refreshing the
   individual service mappings when necessary as specified by the
   Service Boundary. However, when arriving in location B (close to a
   mountain), service sos.mountainrescue, which was not available in
   location A, becomes available. Since the client is only required to
   refresh the mappings for the initially discovered services, the new
   service is not detected. Consequently, the dialstring for the
   mountain-rescue service is not known by the client. Hence, the
   client is unable to recognize an emergency call when the user enters
   the dialstring of the mountain-rescue service, and the emergency call
   may fail altogether.

Note that the Service Boundary (service region for an individual
service) cannot be considered as an indicator for the region for
which a specific Service List is valid. The Service List may even
change within the Service Boundary of another service. For example,
the ambulance mapping is valid for a whole state, but for a part of
the state there is an additional mountain-rescue service.

Consequently, there are two ways to tackle this issue:

o Clients continuously poll for the Service List, although it may
   not have changed.

o The server sends a message containing boundary information that
   tells the client that the Service List does not change inside this
   area.

2. Terminology

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].

3. LoST Extensions

This section describes the necessary extensions to the LoST protocol
in order to support the Service List Boundary in a similar way as the
Service Boundary. Extensions defined in this document are declared
in the new XML namespace urn:ietf:params:xml:ns:lost1:slb.

3.1. Extensions to <listServicesByLocation>

The query <listServicesByLocation> may contain an additional
<serviceListBoundaryRequest> element to additionally request the
boundary for the Service List based on the location provided, with
the resulting location for the list presented either by value or by
reference. In the example below, the value of the 'type' attribute
of the <serviceListBoundaryRequest> element is set to "value":

```
<?xml version="1.0" encoding="UTF-8"?>
<listServicesByLocation
     xmlns="urn:ietf:params:xml:ns:lost1"
     xmlns:gml="http://www.opengis.net/gml"
     xmlns:slb="urn:ietf:params:xml:ns:lost1:slb"
     recursive="true">
  <location id="5415203asdf548" profile="civic">
    <civicAddress xml:lang="en"
       xmlns="urn:ietf:params:xml:ns:pidf:geopriv10:civicAddr">
      <country>AT</country>
      <A1>Lower Austria</A1>
      <A2>Bruck an der Leitha</A2>
      <A3>Wolfsthal</A3>
      <RD>Hauptplatz</RD>
      <HNO>1</HNO>
      <PC>2412</PC>
    </civicAddress>
  </location>
  <service>urn:service:sos</service>
  <slb:serviceListBoundaryRequest type="value"/>
</listServicesByLocation>
```

A <listServicesByLocationResponse> with the addition of one
<serviceListBoundary> element is shown below:

```
<?xml version="1.0" encoding="UTF-8"?>
<listServicesByLocationResponse
     xmlns="urn:ietf:params:xml:ns:lost1"
     xmlns:slb="urn:ietf:params:xml:ns:lost1:slb">
  <serviceList>
    urn:service:sos.ambulance
    urn:service:sos.fire
    urn:service:sos.gas
    urn:service:sos.mountain
    urn:service:sos.poison
    urn:service:sos.police
  </serviceList>
```

```
   <path>
     <via source="resolver.example"/>
     <via source="authoritative.example"/>
   </path>
     <locationUsed id="5415203asdf548"/>
     <slb:serviceListBoundary profile="civic"
       expires="2012-01-01T00:00:00Z">
      <civicAddress xml:lang="en"
         xmlns="urn:ietf:params:xml:ns:pidf:geopriv10:civicAddr">
        <country>AT</country>
        <A1>Lower Austria</A1>
      </civicAddress>
     </slb:serviceListBoundary>
  </listServicesByLocationResponse>
```

The response above indicates that the Service List is valid for Lower
Austria. The <listServicesByLocation> request needs to be repeated
by the client only when moving out of Lower Austria. However, the
mappings of the services themselves may have other service
boundaries. Additionally, the 'expires' attribute indicates the
absolute time when this Service List becomes invalid.

The response MAY contain multiple <serviceListBoundary> elements for
alternative representation, each representing the boundary in a
specific location profile. However, multiple locations inside a
<serviceListBoundary> element are considered to be additive.

The boundary can also be requested by reference when setting the
value of the 'type' attribute of the <serviceListBoundaryRequest>
element to "reference" (which is the default in case the attribute is
omitted). The response will contain a <serviceListBoundaryReference>
element with a 'serviceListKey' attribute (described in Section 3.2),
as shown below.

```
   <?xml version="1.0" encoding="UTF-8"?>
   <listServicesByLocationResponse
        xmlns="urn:ietf:params:xml:ns:lost1"
        xmlns:slb="urn:ietf:params:xml:ns:lost1:slb">
     <serviceList>
        urn:service:sos.ambulance
        urn:service:sos.fire
        urn:service:sos.gas
        urn:service:sos.mountain
        urn:service:sos.poison
        urn:service:sos.police
      </serviceList>
```

```
      <path>
        <via source="resolver.example"/>
        <via source="authoritative.example"/>
      </path>
      <locationUsed id="5415203asdf548"/>
      <slb:serviceListBoundaryReference
         source="authoritative.example"
         serviceListKey="1235678901235678901235678890" />
   </listServicesByLocationResponse>
```

3.2. Retrieving the <serviceListBoundary> via <getServiceListBoundary>

   In order to retrieve the boundary corresponding to a specific
   'serviceListKey', the client issues a <getServiceListBoundary>
   request to the server identified in the 'source' attribute of the
   <serviceListBoundaryReference> element, similar to the
   <getServiceBoundary> request.

   An example is shown below:

```
      <?xml version="1.0" encoding="UTF-8"?>
      <getServiceListBoundary
            xmlns="urn:ietf:params:xml:ns:lost1:slb"
              serviceListKey="1235678901235678901235678890"/>
```

   The LoST server response is shown below:

```
  <?xml version="1.0" encoding="UTF-8"?>
  <getServiceListBoundaryResponse
        xmlns="urn:ietf:params:xml:ns:lost1:slb">
    <serviceListBoundary profile="civic" expires="2012-01-01T00:00:00Z">
      <civicAddress xml:lang="en"
          xmlns="urn:ietf:params:xml:ns:pidf:geopriv10:civicAddr">
        <country>AT</country>
        <A1>Lower Austria</A1>
      </civicAddress>
    </serviceListBoundary>
    <path xmlns="urn:ietf:params:xml:ns:lost1">
      <via source="resolver.example"/>
      <via source="authoritative.example"/>
    </path>
  </getServiceListBoundaryResponse>
```

The 'serviceListKey' uniquely identifies a Service List Boundary, as
the 'key' does for the Service Boundary (see Section 5.6 of
RFC 5222). Therefore, the 'serviceListKey' is a random token with at
least 128 bits of entropy [RFC4086] and can be assumed globally
unique. Whenever the boundary changes, a new 'serviceListKey' MUST
be assigned.

   Note: Since LoST does not define an attribute to indicate which
   location profile the client understands in a <getServiceBoundary>
   request, this document also does not define one for the
   <getServiceListBoundary> request.

3.3. <serviceListBoundary>

For a particular <listServicesByLocation> query, the Service List
Boundary information that gets returned indicates that all the
service identifiers returned in the <serviceList> element are the
same within this geographic region. A Service List Boundary may
consist of geometric shapes (both in civic and geodetic location
format), and may be non-contiguous, like the Service Boundary.

The mapping of the specific services within the Service List Boundary
may be different at different locations.

The server MAY return the boundary information in multiple location
profiles, but MUST use at least one profile that the client used in
the request in order to ensure that the client is able to process the
boundary information.

There is no need to include boundary information in a
<listServicesResponse>. The <listServices> request is purely for
diagnostic purposes and does not contain location information at all,
so boundary information cannot be calculated.

Also note that the Service List Boundary is OPTIONAL, and the LoST
server may return it or not, based on its local policy -- as is the
case with the Service Boundary. However, especially for emergency
services, the Service List Boundary might be crucial to ensure that
moving clients do not miss changes in the available services.

3.4. Implementation Considerations

   The subsections below discuss implementation issues for the LoST
   server and client for Service List Boundary support.

3.4.1. Server Side

   The mapping architecture and framework [RFC5582] states that each
   tree announces its coverage region (for one type of service, e.g.,
   sos.police) to one or more forest guides. Forest guides peer with
   each other and synchronize their data. Hence, a forest guide has
   sufficient knowledge (it knows all the services and their coverage
   regions) to answer a <listServicesByLocation> query and to add the
   <serviceListBoundary> or <serviceListBoundaryReference> as well.

   The calculation of the largest possible area for which the Service
   List stays the same might be a complex task. An alternative would be
   to return smaller areas that are easier to compute. In such a case,
   some unnecessary queries to the LoST server will be a consequence,
   but the main purpose of the Service List Boundary is still achieved:
   to never miss a change of available services. Thus, the server
   operator may specify a reasonable trade-off between the effort to
   generate the boundary information and the saved queries to the LoST
   server.

   For example, in some countries the offered services may differ in
   adjacent counties (or districts, cantons, states, etc.). Their
   borders may be suitable as a Service List Boundary as well, even
   though some adjacent counties offer the same services.

   Other countries might have different structures, and the generation
   of the Service List Boundary might follow other rules as long as it
   is ensured that a client is able to notice any change in the Service
   List when moving.

3.4.2. Client Side

   A mobile client that already implements LoST and evaluates the
   <serviceBoundary> has almost everything that is needed to make use of
   the Service List Boundary. Since the integration into LoST follows
   the concept of the <serviceBoundary> (and also makes use of the same
   location profiles), only the additional <serviceListBoundary> needs
   to be evaluated. Whenever moving outside a Service List Boundary,
   the client performs a new <listServicesByLocation> query with the new
   location information in order to determine a change in available
   services.

4. Security and Privacy Considerations

   Security considerations for LoST are discussed in [RFC5222]. This
   document extends LoST to also carry Service List Boundaries (and
   requests for them). These Service List Boundaries are calculated by
   the server based on the individual Service Boundaries and sent to
   clients in case the local policy allows this. Therefore, it is
   generally considered to have the same level of sensitivity as for the
   Service Boundary and thus the same access control and confidentiality
   requirements as the base LoST protocol. As a result, the security
   measures incorporated in the base LoST specification [RFC5222]
   provide sufficient protection for LoST messages that use the Service
   List Boundary extension.

5. IANA Considerations

   IANA has taken two actions: an XML schema registration and a
   namespace registration, according to the description in the following
   sections.

5.1. Relax NG Schema Registration

   IANA has registered the following Relax NG Schema in the IETF XML
   Registry [RFC3688]:

   URI: urn:ietf:params:xml:schema:lost1:slb

   Registrant Contact: IETF ECRIT Working Group, Karl Heinz Wolf
   (karlheinz.wolf@nic.at)

   Relax NG Schema:

   BEGIN

   <?xml version="1.0" encoding="UTF-8"?>
   <grammar
       xmlns="http://relaxng.org/ns/structure/1.0"
       xmlns:a="http://relaxng.org/ns/compatibility/annotations/1.0"
       xmlns:slb="urn:ietf:params:xml:ns:lost1:slb"
       ns="urn:ietf:params:xml:ns:lost1"
       datatypeLibrary="http://www.w3.org/2001/XMLSchema-datatypes">
     <include href="lost.rng">

```
    <!-- redefinition of LoST elements -->
      <start>
        <choice>
          <ref name="findService"/>
          <ref name="listServices"/>
          <ref name="listServicesByLocation"/>
          <ref name="getServiceBoundary"/>
          <ref name="findServiceResponse"/>
          <ref name="listServicesResponse"/>
          <ref name="listServicesByLocationResponse"/>
          <ref name="getServiceBoundaryResponse"/>
          <ref name="errors"/>
          <ref name="redirect"/>

          <!-- New in RFC 6197 -->
          <ref name="getServiceListBoundary"/>
          <ref name="getServiceListBoundaryResponse"/>
        </choice>
      </start>

      <define name="listServicesByLocation">
        <element name="listServicesByLocation">
          <ref name="requestLocation"/>
          <ref name="commonRequestPattern"/>
          <optional>
            <attribute name="recursive">
              <data type="boolean"/>
              <a:defaultValue>true</a:defaultValue>
            </attribute>
          </optional>

          <!-- New in RFC 6197 -->
          <optional>
            <ref name="serviceListBoundaryRequest"/>
          </optional>
        </element>
      </define>

      <define name="listServicesByLocationResponse">
        <element name="listServicesByLocationResponse">
          <ref name="serviceList"/>
          <ref name="commonResponsePattern"/>
          <ref name="locationUsed"/>
```

```
            <!-- New in RFC 6197 -->
            <optional>
              <choice>
                <ref name="serviceListBoundary"/>
                <ref name="serviceListBoundaryReference"/>
              </choice>
            </optional>
          </element>
        </define>
      </include>

      <define name="serviceListBoundaryRequest">
        <element name="slb:serviceListBoundaryRequest">
          <optional>
            <attribute name="type">
              <choice>
                <value>value</value>
                <value>reference</value>
              </choice>
              <a:defaultValue>reference</a:defaultValue>
            </attribute>
          </optional>
        </element>
      </define>

      <define name="serviceListBoundary">
       <oneOrMore>
        <element name="slb:serviceListBoundary">
          <optional>
            <ref name="expires"/>
          </optional>
          <ref name="locationInformation"/>
          <ref name="extensionPoint"/>
        </element>
       </oneOrMore>
      </define>

      <define name="serviceListBoundaryReference">
        <element name="slb:serviceListBoundaryReference">
          <ref name="source"/>
          <attribute name="serviceListKey">
            <data type="token"/>
          </attribute>
        <ref name="extensionPoint"/>
        </element>
      </define>
```

```
   <define name="getServiceListBoundary">
     <element name="slb:getServiceListBoundary">
       <attribute name="serviceListKey">
         <data type="token"/>
       </attribute>
     <ref name="extensionPoint"/>
     </element>
   </define>

   <define name="getServiceListBoundaryResponse">
     <element name="slb:getServiceListBoundaryResponse">
      <ref name="serviceListBoundary"/>
      <ref name="path"/>
      <ref name="extensionPoint"/>
     </element>
   </define>
 </grammar>

 END
```

5.2. Namespace Registration

   IANA has registered the following namespace (below the LoST namespace
   defined in [RFC5222]) in the IETF XML Registry [RFC3688]:

   URI: urn:ietf:params:xml:ns:lost1:slb

   Registrant Contact: IETF ECRIT Working Group, Karl Heinz Wolf
   (karlheinz.wolf@nic.at)

   XML:

   BEGIN

   <?xml version="1.0"?>
   <!DOCTYPE html PUBLIC "-//W3C//DTD XHTML Basic 1.0//EN"
      "http://www.w3.org/TR/xhtml-basic/xhtml-basic10.dtd">
   <html xmlns="http://www.w3.org/1999/xhtml">
   <head>
     <meta http-equiv="content-type"
           content="text/html;charset=iso-8859-1"/>
     <title>LoST Service List Boundary Namespace</title>
   </head>
   <body>
     <h1>Namespace for the LoST Service List Boundary</h1>
     <h2>urn:ietf:params:xml:ns:lost1:slb</h2>
   <p>See
     RFC 6197.</p>
   </body>
   </html>

   END

6. Acknowledgements

   The author would like to thank Henning Schulzrinne for discussion of
   the document, and Martin Thomson, Richard Barnes, and Roger Marshall
   for their valuable input and text suggestions during the working
   group Last Call. Further thanks go to Joshua Bell from the
   Applications Area Review Team for his help with Relax NG.

7. References

7.1. Normative References

   [RFC5222] Hardie, T., Newton, A., Schulzrinne, H., and H.
              Tschofenig, "LoST: A Location-to-Service Translation
              Protocol", RFC 5222, August 2008.

   [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.

   [RFC4086] Eastlake 3rd, D., Schiller, J., and S. Crocker,
              "Randomness Requirements for Security", BCP 106, RFC 4086,
              June 2005.

7.2. Informative References

   [RFC5582] Schulzrinne, H., "Location-to-URL Mapping Architecture and
              Framework", RFC 5582, September 2009.

Author's Address

   Karl Heinz Wolf
   nic.at GmbH
   Karlsplatz 1/2/9
   Wien A-1010
   Austria

   Phone: +43 1 5056416 37
   EMail: karlheinz.wolf@nic.at
   URI: http://www.nic.at/