ECRIT
Internet-Draft
Expires: April 9, 2010

K. Wolf
nic.at

October 6, 2009

Location-to-Service Translation Protocol (LoST) Extension: ServiceListBoundary
draft-ietf-ecrit-lost-servicelistboundary-00

Status of this Memo

Copyright Notice

Abstract

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 response from the LoST server does not provide information about the geographical region for which the returned service list is valid. Therefore, this document proposes a ServiceListBoundary to assist the client to not miss a change in available services when moving.

Table of Contents

1. Introduction

Location based service providers as well as Public Safety Answering Points (PSAPs) only serve a specific geographic region. Therefore the LoST protocol defines the ServiceBoundary, 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 by the <listServicesByLocation> query in LoST [RFC5222]. The LoST server returns a list of services that are available at this particular location. But the server does not inform the client for which geographical region the returned service list is valid. This may lead to the situation where a client initially discovered all available services by the <listServicesByLocation> query, and then moves to a different location while refreshing the service mappings, but does not notice the availability of another service. 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 services list:

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 told by the ServiceBoundary. However, when arriving in location B (close to a mountain), service sos.mountainrescue is available, which was not available in location A. Nevertheless, the client does not detect this, because only the mapping of the initially discovered services (police, ambulance, fire) are refreshed. Consequently, the dialstring for the mountain rescue is not known by the client, and the emergency call to the mountain rescue service will certainly fail.

Note that the ServiceBoundary (service region for an individual service) cannot be considered as an indicator for the region a specific service list is valid for. The service list may even change within the ServiceBoundary 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:

- clients continuously ask for the service list, although it may not have changed
- a boundary information (telling the client that the service list does not change inside this area)

Since the LoST protocol has the ServiceBoundary concept in order to avoid that clients continuously try to refresh the mapping of a specific service, a ServiceListBoundary would provide a similar mechanism for service lists.

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 chapter describes the necessary modifications to the LoST protocol in order to support the proposed ServiceListBoundary in a similar way as the ServiceBoundary.

3.1. Extensions to <ListServiceByLocation>

The query <listServicesByLocation> may contain an additional serviceListBoundary element to request the boundary for the service list, either by value or by reference. In the example below the value of the serviceListBoundary element ist 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="TBD"
  recursive="true">
  <location id="mylocation" profile="civic">
    <civicAddress
      xmlns="urn:ietf:params:xml:ns:pidf:geopriv10:civicAddr">
      <country>AT</country>
      <A1>Lower Austria</A1>
      <A3>Wolfsthal</A3>
      <RD>Hauptplatz</RD>
      <HNO>1</HNO>
      <PC>2412</PC>
    </civicAddress>
  </location>
  <service>urn:service:sos</service>
  <slb:serviceListBoundary>value</slb:serviceListBoundary>
</listServicesByLocation>
```

A possible response is shown below:

```
<?xml version="1.0" encoding="UTF-8"?>
<listServicesByLocationResponse
 xmlns="urn:ietf:params:xml:ns:lost1">
 xmlns:slb="TBD"
 <serviceList expires="2010-01-01T00:00:00Z">
  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="mylocation"/>
 <slb:serviceListBoundary profile="civic">
   <civicAddress
     xmlns="urn:ietf:params:xml:ns:pidf:geopriv10:civicAddr">
     <country>AT</country>
     <A1>Lower Austria</A1>
   </civicAddress>
 </slb:serviceListBoundary>
</listServicesByLocationResponse>
```

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

The boundary can also be requested by reference when setting the attribute serviceListBoundary to "reference". Then the response contains a serviceListBoundaryReference element, as shown below.

```
<?xml version="1.0" encoding="UTF-8"?>
<listServicesByLocationResponse
 xmlns="urn:ietf:params:xml:ns:lost1">
 xmlns:slb="TBD"
 <serviceList expires="2010-01-01T00:00:00Z">
  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="mylocation"/>
 <serviceListBoundaryReference
    source="authoritative.example"
    serviceListKey="1235678901235678901235678901" />
</listServicesByLocationResponse>
```

3.2. Retrieving the serviceList Boundary via getServiceListBoundary

In order to retrieve the boundary corresponding a specific serviceListKey, the client issues a <getServiceListBoundary> request, 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"
    serviceListKey="1235678901235678901235678901"/>
```

The LoST server response is shown below:

```
<?xml version="1.0" encoding="UTF-8"?>
<getServiceListBoundaryResponse xmlns="TBD">
  <serviceListBoundary profile="civic" expires="2010-01-01T00:00:00Z">
    <civicAddress
      xmlns="urn:ietf:params:xml:ns:pidf:geopriv10:civicAddr">
     <country>AT</country>
     <A1>Lower Austria</A1>
    </civicAddress>
  </serviceListBoundary>
  <path>
    <via source="resolver.example"/>
    <via source="authoritative.example"/>
  </path>
</getServiceListBoundaryResponse>
```

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

3.3. Service List Boundary

The service list boundary indicates a region within which all <listServicesByLocation> queries with the same service identifiers result in the same serviceList. 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 profiles, but has to 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.

TBD: For <getServiceListBoundary> an attribute in the request could be used to indicate which profile the client understands (e.g. <getServiceListBoundary profile="civic"...)

There is no need to include boundary information to a <listServicesResponse>. <ListServices> requests are purely for diagnostic purposes and do not contain location information at all, so no boundary information is reasonable.

Also note that the serviceListBoundary is optional and the LoST

server may return it based on its local policy - like it is the case with service boundary. However, especially for emergency services, the serviceListBoundary might be crucial to ensure that moving clients do not miss changes in the available services.

3.4. Implementation Considerations

The subsections below discuss implementations issues for the LoST server and client for the serviceListBoundary support.

3.4.1. Server Side

The mapping architecture and framework [RFC5582] describes 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 additionally add the serviceListBoundary 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 unneeded queries to the LoST server are the consequence, but still the main purpose of the serviceListBoundary is achieved: Never miss a change of available services. So a reasonable trade-off between the effort to generate the boundary information and the saved queries to the LoST server has to be considered.

Probably for some countries the county (or disrict, canton, state, ...) borders would be suitable as serviceListBoundary. Some neighbouring counties may have implemented different services while a listServicesByLocation query in other neighbouring counties still results in the same serviceList. So when moving across a county border, it is at least ensured, that every device fetches a new service list from the LoST server.

Other countries might have different structures and the generation of the serviceListBoundary 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 serviceListBoundary. Since the integration into LoST follows the concept of the serviceBoundary (and also makes use of the same

location profiles), just the additional serviceListBoundary has to be evaluated. Whenever moving outside a serviceListBoundary, the client must perform a new listServicesByLocation query with the new location information in order to determine a change in available services.

4. Security & Privacy Considerations

Security considerations are discussed in [RFC5222].

5. IANA Considerations

TODO.

6. Acknowledgement

The author would like to thank Henning Schulzrinne for the discussion on the draft.

7. Normative References

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

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

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

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/