

ECRIT
Internet-Draft
Intended status: Experimental
Expires: July 16, 2012

H. Schulzrinne
Columbia University
H. Tschofenig
Nokia Siemens Networks
January 13, 2012

**Synchronizing Location-to-Service Translation (LoST) Protocol based
Service Boundaries and Mapping Elements
draft-ietf-ecrit-lost-sync-16.txt**

Abstract

The Location-to-Service Translation (LoST) protocol is an XML-based protocol for mapping service identifiers and geodetic or civic location information to service URIs and service boundaries. In particular, it can be used to determine the location-appropriate Public Safety Answering Point (PSAP) for emergency services.

The main data structure, the <mapping> element, used for encapsulating information about service boundaries is defined in the LoST protocol specification and circumscribes the region within which all locations map to the same service Uniform Resource Identifier (URI) or set of URIs for a given service.

This document defines an XML protocol to exchange these mappings between two nodes. This mechanism is designed for the exchange of authoritative <mapping> elements between two entities. Exchanging cached <mapping> elements, i.e. non-authoritative elements, is possible but not envisioned. In any case, this document can also be used without the LoST protocol even though the format of the <mapping> element is re-used from the LoST specification.

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 July 16, 2012.

Copyright Notice

Copyright (c) 2012 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to [BCP 78](#) and the IETF Trust's Legal Provisions Relating to IETF Documents (<http://trustee.ietf.org/license-info>) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

Table of Contents

1.	Introduction	4
2.	Terminology	10
3.	Querying for Mappings with a <getMappingsRequest> / <getMappingsResponse> Exchange	11
3.1.	Behavior of the LoST Sync Destination	11
3.2.	Behavior of the LoST Sync Source	11
3.3.	Examples	12
4.	Pushing Mappings via <pushMappings> and <pushMappingsResponse>	14
4.1.	Behavior of the LoST Sync Source	14
4.2.	Behavior of the LoST Sync Destination	14
4.3.	Example	15
5.	Transport	19
6.	RelaxNG	20
7.	Operational Considerations	22
8.	Security Considerations	23
9.	IANA Considerations	24
9.1.	Content-type registration for 'application/lostsync+xml'	24
9.2.	LoST Sync Relax NG Schema Registration	25
9.3.	LoST Synchronization Namespace Registration	26
10.	Acknowledgments	27
11.	References	28
11.1.	Normative References	28
11.2.	Informative References	28
	Authors' Addresses	29

1. Introduction

The LoST (Location-to-Service Translation) protocol [[RFC5222](#)] maps service identifiers and geodetic or civic location information to service URIs. The main data structure, the <mapping> element, used for encapsulating information about service boundaries is defined in the LoST protocol specification and circumscribes the region within which all locations map to the same service Uniform Resource Identifier (URI) or set of URIs for a given service.

This mechanism is designed for the exchange of authoritative <mapping> elements between two entities (the LoST Sync source and the LoST Sync destination).

The LoST Sync mechanism can, for example, be used in the LoST architecture, as specified in the [[RFC5582](#)]. There, LoST servers act in different roles that cooperate to provide an ubiquitous, globally scalable and resilient mapping service. In the LoST mapping architecture, LoST servers can peer, i.e., have an on-going data exchange relationship. Peering relationships are set up manually, based on local policies. A LoST server may peer with any number of other LoST servers. Forest guides peer with other forest guides; authoritative mapping servers peer with forest guides and other authoritative servers, either in the same cluster or above or below them in the tree. Authoritative mapping servers push coverage regions "up" the tree, i.e., from child nodes to parent nodes. The child informs the parent of the geospatial or civic region that it covers for a specific service.

Consider a hypothetical deployment of LoST in two countries, for example Austria and Finland. Austria, in our example, runs three authoritative mapping servers labeled as 'East', 'West' and 'Vienna' whereby the former two cover the entire country except for Vienna, which is covered by a separate LoST server. There may be other caching LoST servers run by ISPs, universities, and VSPs but they are not relevant for this illustration. Finland, on the other hand, decided to only deploy a single LoST server that also acts as a Forest Guide. For this simplistic illustration we assume that only one service is available, namely 'urn:service:sos' since otherwise the number of stored mappings would have to be multiplied by the number of used services.

Figure 1 shows the example deployment.

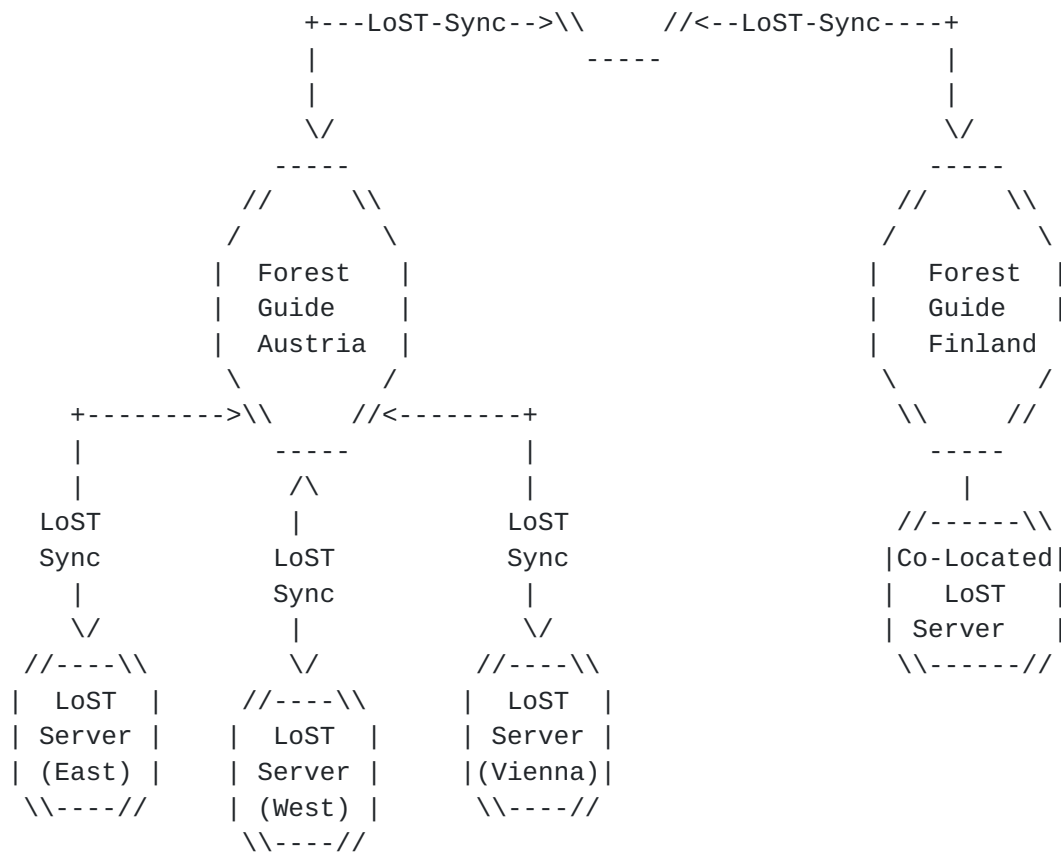


Figure 1: LoST Deployment Example

The configuration of these nodes would therefore be as follows:

Forest Guide Austria: This forest guide would contain mappings for the three authoritative mapping servers (East, West and Vienna) describing what area they are responsible for. Note that each mapping would contain a service URN and these mappings point to LoST servers rather than to PSAPs or ESRPs.

LOST Server 'East': This LoST server would contain all the mappings to PSAPs covering one half of the country.

Additionally, the LoST server aggregates all the information it has and provides an abstracted view towards the Forest Guide indicating that it is responsible for a certain area (for a given service, and for a given location profile). Such a mapping could have the following structure:


```

<?xml version="1.0" encoding="UTF-8"?>
<mapping
  xmlns="urn:ietf:params:xml:ns:lost1"
  xmlns:gml="http://www.opengis.net/gml"
  expires="2009-01-01T01:44:33Z"
  lastUpdated="2009-12-01T01:00:00Z"
  source="east-austria.lost-example.com"
  sourceId="e8b05a41d8d1415b80f2cddb96ccf109">
  <displayName xml:lang="en">LoST Server 'East'</displayName>
  <service>urn:service:sos</service>
  <serviceBoundary profile="geodetic-2d">
    <gml:Polygon srsName="urn:ogc:def::crs:EPSG::4326">
      <gml:exterior>
        <gml:LinearRing>
          <gml:pos> ... </gml:pos>
          ..... list of coordinates for
            boundary of LoST server 'East'
          <gml:pos> ... </gml:pos>
        </gml:LinearRing>
      </gml:exterior>
    </gml:Polygon>
  </serviceBoundary>
  <uri/>
</mapping>

```

Figure 2: Forest Guide Austria Mapping XML Snippet

Note that the XML code snippet in Figure 2 serves illustrative purposes only and does not validate. As it can be seen in this example the <uri> element is absent and the 'source' attribute identifies the LoST server, namely "east-austria.lost-example.com".

The above-shown mapping is what is the LoST server "east-austria.lost-example.com" provides to the Austrian Forest Guide.

LoST Server 'West': This LoST server would contain all the mappings to PSAPs covering the other half of the country.

LoST Server 'Vienna': This LoST server would contain all the mappings to PSAPs in the area of Vienna.

Forest Guide Finland: In our example we assume that Finland would deploy a single ESRP for the entire country as their IP-based emergency services solution. There is only a single LoST server and it is co-located with the Forest Guide, as shown in Figure 1. The mapping data this FG would distribute via LoST sync is shown in Figure 3.


```

<?xml version="1.0" encoding="UTF-8"?>
<mapping xmlns="urn:ietf:params:xml:ns:lost1"
  expires="2007-01-01T01:44:33Z"
  lastUpdated="2006-11-01T01:00:00Z"
  source="finland.lost-example.com"
  sourceId="7e3f40b098c711dbb6060800200c9a66">
  <displayName xml:lang="en">Finland ESRP</displayName>
  <service>urn:service:sos</service>
  <serviceBoundary profile="civic">
    <civicAddress
      xmlns="urn:ietf:params:xml:ns:pidf:geopriv10:civicAddr">
      <country>FI</country>
    </civicAddress>
  </serviceBoundary>
  <uri/>
</mapping>

```

Figure 3: Forest Guide Finland Mapping Example

An example mapping stored at the co-located LoST server is shown in Figure 4.

```

<?xml version="1.0" encoding="UTF-8"?>
<mapping xmlns="urn:ietf:params:xml:ns:lost1"
  expires="2007-01-01T01:44:33Z"
  lastUpdated="2006-11-01T01:00:00Z"
  source="finland.lost-example.com"
  sourceId="7e3f40b098c711dbb6060800200c9a66">
  <displayName xml:lang="en">Finland ESRP</displayName>
  <service>urn:service:sos</service>
  <serviceBoundary profile="civic">
    <civicAddress
      xmlns="urn:ietf:params:xml:ns:pidf:geopriv10:civicAddr">
      <country>FI</country>
    </civicAddress>
  </serviceBoundary>
  <uri>sip:esrp@finland-example.com</uri>
  <uri>xmpp:esrp@finland-example.com</uri>
  <serviceNumber>112</serviceNumber>
</mapping>

```

Figure 4: Forest Guide Finland / Co-Located LoST Server Mapping Example

The LoST sync mechanism described in this document could be run between the two Forest Guides. Thereby, the three mappings stored in the Austria FG are sent to the FG Finland and a single mapping in the FG Finland is sent to the FG Austria. Additionally, the three

Austrian LoST servers could utilize LoST sync to inform the Austrian FG about their boundaries. These three authoritative mapping servers in Austria would be responsible to maintain their own mapping information. Since the amount of data being exchanged is small and the expected rate of change is low the nodes are configured to always exchange all their mapping information whenever a change happens.

This document defines two types of exchanges and those are best described by the exchange between two nodes as shown in Figure 5 and Figure 6. The protocol exchange always runs between a LoST Sync source and a LoST Sync destination. Node A in the examples of Figure 5 and Figure 6 has mappings that Node B is going to retrieve. Node A acts as the source for the data and Node B is the destination.

The `<getMappingsRequest>` request allows a LoST Sync source to request mappings from a LoST Sync destination.

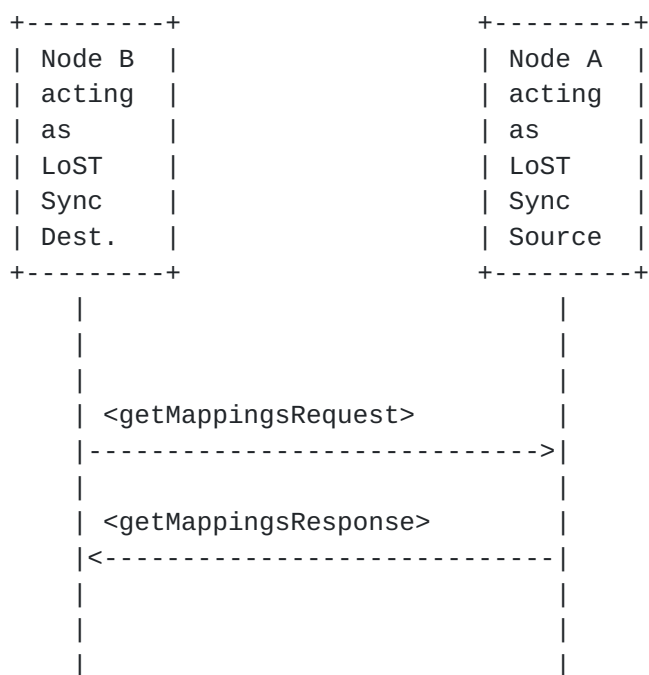


Figure 5: Querying for Mappings with a `<getMappingsRequest>` Message

Note that in the exchange illustrated in Figure 5 Node B is issuing the first request and plays the role of the HTTPS client and Node A plays the role of the HTTPS server.

The `<pushMappingsRequest>` exchange allows a LoST Sync source to push mappings to LoST Sync destination. The assumption is being made that Node A and B have previously been configured in a way that they push mappings in such a fashion and that Node A maintains state about the mappings have to be pushed to Node B. No subscribe mechanism is

defined in this document that would allow Node B to tell Node A about what mappings it is interested nor a mechanism for learning to which entities mappings have to be pushed.

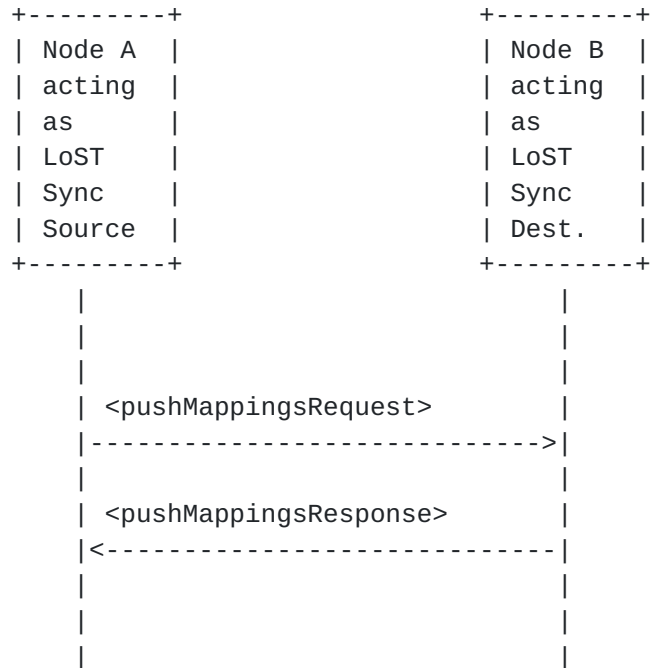


Figure 6: Pushing Mappings with a <pushMappingsRequest> Message

Note that in the exchange illustrated in Figure 6 Node A issuing the first request and plays the role of the HTTPS client and Node B plays the role of the HTTPS server.

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](#)].

This document reuses terminology introduced by the mapping architecture document [[RFC5582](#)], such as 'coverage region', 'forest guide', 'mapping', 'authoritative mapping server', etc..

Throughout this document we use the term LoST Sync source and LoST Sync destination to denote the protocol end points of the exchange. The protocol is referred as LoST Sync within the text.

3. Querying for Mappings with a <getMappingsRequest> / <getMappingsResponse> Exchange

3.1. Behavior of the LoST Sync Destination

A LoST Sync destination has two ways to retrieve mapping elements from a LoST Sync source.

1. A mechanisms that is suitable when no mappings are available on the LoST Sync destination is to submit an empty <getMappingsRequest> message, as shown in Figure 7. The intent by the LoST Sync destination thereby is to retrieve all mappings from the LoST Sync source. Note that the request does not propagate further to other nodes.
2. In case a LoST Sync destination node has already obtained mappings in previous exchanges then it may want to check whether these mappings have been updated in the meanwhile. The policy when to poll for updated mapping information is outside the scope of this document. The <getMappingsRequest> message with one or multiple <exists> child element(s) allows to reduce the number of returned mappings to those that have been updated and also to those that are missing.

In response to the <getMappingsRequest> message the LoST Sync destination waits for the <getMappingsResponse> message. In case of a successful response the LoST Sync destination stores the received mappings and determines which mappings to update.

3.2. Behavior of the LoST Sync Source

When a LoST Sync source receives an empty <getMappingsRequest> message then all locally available mappings MUST be returned.

When a LoST Sync source receives a <getMappingsRequest> message with one or multiple <exists> child element(s) then it MUST consult with the local mapping database to determine whether any of the mappings of the client is stale and whether there are mappings locally that the client does not yet have. The former can be determined by finding mappings corresponding to the 'source' and 'sourceID' attribut where a mapping with a more recent lastUpdated date exists.

Processing a <getMappingsRequest> message MAY lead to a successful response in the form of a <getMappingsResponse> or an <errors> message. Only the <badRequest>, <forbidden>, <internalError>, <serverTimeout> errors, defined in [RFC5222], are utilized by this specification. Neither the <redirect> nor the <warnings> messages are reused by this message.

3.3. Examples

The first example shows an empty `<getMappingsRequest>` message that would retrieve all locally stored mappings at the LoST Sync source.

```
<?xml version="1.0" encoding="UTF-8"?>
<getMappingsRequest xmlns="urn:ietf:params:xml:ns:lostsync1"/>
```

Figure 7: Example of empty `<getMappingsRequest>` message

A further example request is shown in Figure 8 and the corresponding response is depicted in Figure 9. In this example the `<getMappingsRequest>` element contains information about the mapping that is locally available to the client inside the `<mapping-fingerprint>` element (with `source="authoritative.bar.example"`, `sourceId="7e3f40b098c711dbb6060800200c9a66"`, and `lastUpdated="2006-11-01T01:00:00Z"`). The query asks for mappings that are more recent than the available one as well as any missing mapping.

```
<?xml version="1.0" encoding="UTF-8"?>
<getMappingsRequest xmlns="urn:ietf:params:xml:ns:lostsync1">
  <exists>
    <mapping-fingerprint source="authoritative.bar.example"
      sourceId="7e3f40b098c711dbb6060800200c9a66"
      lastUpdated="2006-11-01T01:00:00Z">
    </mapping-fingerprint>
  </exists>
</getMappingsRequest>
```

Figure 8: Example `<getMappingsRequest>` Message

The response to the above request is shown in Figure 9. A more recent mapping was available with the identification of `source="authoritative.bar.example"` and `sourceId="7e3f40b098c711dbb6060800200c9a66"`. Only one mapping that matched `source="authoritative.foo.example"` was found and returned.

```
<?xml version="1.0" encoding="UTF-8"?>
<sync:getMappingsResponse
  xmlns:sync="urn:ietf:params:xml:ns:lostsync1"
  xmlns="urn:ietf:params:xml:ns:lost1"
  xmlns:gml="http://www.opengis.net/gml">

  <mapping source="authoritative.bar.example"
    sourceId="7e3f40b098c711dbb6060800200c9a66"
```



```
      lastUpdated="2008-11-26T01:00:00Z"
      expires="2009-12-26T01:00:00Z">
      <displayName xml:lang="en">Leonia Police Department
    </displayName>
    <service>urn:service:sos.police</service>
    <serviceBoundary
profile="urn:ietf:params:lost:location-profile:basic-civic">
      <civicAddress
xmlns="urn:ietf:params:xml:ns:pidf:geopriv10:civicAddr">
        <country>US</country>
        <A1>NJ</A1>
        <A3>Leonia</A3>
        <PC>07605</PC>
      </civicAddress>
    </serviceBoundary>
    <uri>sip:police@leonianj2.example.org</uri>
    <serviceNumber>911</serviceNumber>
  </mapping>

  <mapping expires="2009-01-01T01:44:33Z"
    lastUpdated="2008-11-01T01:00:00Z"
    source="authoritative.foo.example"
    sourceId="7e3f40b098c711dbb6060111111111111">
    <displayName xml:lang="en">New York City Police Department
  </displayName>
  <service>urn:service:sos.police</service>
  <serviceBoundary profile="geodetic-2d">
    <gml:Polygon srsName="urn:ogc:def::crs:EPSG::4326">
      <gml:exterior>
        <gml:LinearRing>
          <gml:pos>37.775 -122.4194</gml:pos>
          <gml:pos>37.555 -122.4194</gml:pos>
          <gml:pos>37.555 -122.4264</gml:pos>
          <gml:pos>37.775 -122.4264</gml:pos>
          <gml:pos>37.775 -122.4194</gml:pos>
        </gml:LinearRing>
      </gml:exterior>
    </gml:Polygon>
  </serviceBoundary>
  <uri>sip:nypd@example.com</uri>
  <uri>xmpp:nypd@example.com</uri>
  <serviceNumber>911</serviceNumber>
</mapping>

</sync:getMappingsResponse>
```

Figure 9: Example <getMappingsResponse> Message

4. Pushing Mappings via <pushMappings> and <pushMappingsResponse>

4.1. Behavior of the LoST Sync Source

When a LoST Sync source obtains new information that is of interest to its peers, it may push the new mappings to its peers. Configuration settings at both peers decide whether this functionality is used and what mappings are pushed to which other peers. New mappings may arrive through various means, such as a manual addition to the local mapping database, or through the interaction with other entities. Deleting mappings may also trigger a protocol interaction.

The LoST Sync source SHOULD keep track to which LoST Sync destination it has pushed mapping elements. If it does not keep state information then it always has to push the complete data set. As discussed in [Section 5.1 of \[RFC5222\]](#), mapping elements are identified by the 'source', 'sourceID' and 'lastUpdated' attributes. A mapping is considered the same if these three attributes match. It is RECOMMENDED not to push the same information to the same peer more than once.

A <pushMappings> request sent by a LoST Sync source MUST containing one or more <mapping> elements.

To delete a mapping, the content of the mapping is left empty, i.e. the <mapping> element only contains the 'source', 'sourceID', 'lastUpdated', and 'expires' attribute. Figure 10 shows an example request where the mapping with the source="nj.us.example", sourceId="123", lastUpdated="2008-11-01T01:00:00Z", expires="2008-11-01T01:00:00Z" is requested to be deleted. Note that the 'expires' attribute is required per schema definition but will be ignored in processing the request on the receiving side. A sync source may want to delete the mapping from its internal mapping database, but has to remember which peers it has distributed this update to unless it has other ways to ensure that databases do not get out of sync.

4.2. Behavior of the LoST Sync Destination

When a LoST Sync destination receives a <pushMappingsRequest> message then the cache with the existing mappings is inspected to determine whether the received mapping should lead to an update of an already existing mapping, should create a new mapping in the cache, or should be discarded.

A newly received mapping MUST update an existing one having the same 'source' and 'sourceId' and a more recent time in 'lastUpdated'.

If the received mapping does not match with any existing mapping based on the 'source' and 'sourceId' then it MUST be added to the local cache as an independent mapping.

If a <pushMappingsRequest> message with an empty <mapping> element is received then a corresponding mapping has to be determined based on the 'source', and the 'sourceID'.

If no mapping can be identified then an <errors> response MUST be returned that contains the <notDeleted> child element. The <notDeleted> element MAY contain a 'message' attribute with an error description used for debugging purposes. The <notDeleted> element MUST contain the <mapping> element(s) that caused the error.

The response to a <pushMappingsRequest> request is a <pushMappingsResponse> message. With this specification, a successful response message returns no additional elements, whereas an <errors> response is returned in the response message, if the request failed. Only the <badRequest>, <forbidden>, <internalError> or <serverTimeout> errors defined in [Section 13.1 of \[RFC5222\]](#), are used. The <redirect> and <warnings> messages are not used for this query/response.

If the set of nodes that are synchronizing their data does not form a tree, it is possible that the same information arrives through several other nodes. This is unavoidable, but generally only imposes a modest overhead. (It would be possible to create a spanning tree in the same fashion as IP multicast, but the complexity does not seem warranted, given the relatively low volume of data.)

4.3. Example

An example is shown in Figure 10. Imagine a LoST node that obtained two new mappings identified as follows:

o

```
source="authoritative.example"
sourceId="7e3f40b098c711dbb6060800200c9a66"
lastUpdated="2008-11-26T01:00:00Z"
```

o

```
source="authoritative.example"
sourceId="7e3f40b098c711dbb6060111111111111"
lastUpdated="2008-11-01T01:00:00Z"
```

These two mappings have to be added to the peer's mapping database.

Additionally, the following mapping has to be deleted:

- o source="nj.us.example" sourceId="123" lastUpdated="2008-11-01T01:00:00Z"

```
<?xml version="1.0" encoding="UTF-8"?>
<sync:pushMappings
  xmlns:sync="urn:ietf:params:xml:ns:lostsync1"
  xmlns="urn:ietf:params:xml:ns:lost1"
  xmlns:gml="http://www.opengis.net/gml">

  <mapping source="authoritative.example"
    sourceId="7e3f40b098c711dbb6060800200c9a66"
    lastUpdated="2008-11-26T01:00:00Z"
    expires="2009-12-26T01:00:00Z">
    <displayName xml:lang="en">Leonía Police Department
    </displayName>
    <service>urn:service:sos.police</service>
    <serviceBoundary
profile="urn:ietf:params:lost:location-profile:basic-civic">
      <civicAddress
xmlns="urn:ietf:params:xml:ns:pidf:geopriv10:civicAddr">
        <country>US</country>
        <A1>NJ</A1>
        <A3>Leonía</A3>
        <PC>07605</PC>
      </civicAddress>
    </serviceBoundary>
    <uri>sip:police@leonianj.example.org</uri>
    <serviceNumber>911</serviceNumber>
  </mapping>

  <mapping expires="2009-01-01T01:44:33Z"
    lastUpdated="2008-11-01T01:00:00Z"
    source="authoritative.example"
    sourceId="7e3f40b098c711dbb606011111111111">
    <displayName xml:lang="en">New York City Police Department
    </displayName>
    <service>urn:service:sos.police</service>
    <serviceBoundary profile="geodetic-2d">
      <gml:Polygon srsName="urn:ogc:def::crs:EPSG::4326">
        <gml:exterior>
          <gml:LinearRing>
            <gml:pos>37.775 -122.4194</gml:pos>
            <gml:pos>37.555 -122.4194</gml:pos>
            <gml:pos>37.555 -122.4264</gml:pos>
            <gml:pos>37.775 -122.4264</gml:pos>
```



```
        <gml:pos>37.775 -122.4194</gml:pos>
      </gml:LinearRing>
    </gml:exterior>
  </gml:Polygon>
</serviceBoundary>
<uri>sip:nypd@example.com</uri>
<uri>xmpp:nypd@example.com</uri>
<serviceNumber>911</serviceNumber>
</mapping>

<mapping source="nj.us.example"
  sourceId="123"
  lastUpdated="2008-11-01T01:00:00Z"
  expires="2008-11-01T01:00:00Z"/>

</sync:pushMappings>
```

Figure 10: Example <pushMappingsRequest> Message

In response, the peer performs the necessary operation and updates its mapping database. In particular, it will check whether the other peer is authorized to perform the update and whether the elements and attributes contain values that it understands. In our example, a positive response is returned as shown in Figure 11.

```
<?xml version="1.0" encoding="UTF-8"?>
<pushMappingsResponse xmlns="urn:ietf:params:xml:ns:lostsync1" />
```

Figure 11: Example <pushMappingsResponse>

In case that a mapping could not be deleted as requested the following error response might be returned instead.


```
<?xml version="1.0" encoding="UTF-8"?>
<errors xmlns="urn:ietf:params:xml:ns:lost1"
  xmlns:sync="urn:ietf:params:xml:ns:lostsync1"
  source="nodeA.example.com">

  <sync:notDeleted
    message="Could not delete the indicated mapping."
    xml:lang="en">

    <mapping source="nj.us.example"
      sourceId="123"
      lastUpdated="2008-11-01T01:00:00Z"
      expires="2008-11-01T01:00:00Z"/>
  </sync:notDeleted>
</errors>
```

Figure 12: Example <errors> Message

5. Transport

LoST Sync needs an underlying protocol transport mechanism to carry requests and responses. This document uses HTTPS as a transport to exchange XML documents. No fallback to HTTP is provided.

When using HTTP-over-TLS [[RFC2818](#)], LoST Sync messages use the POST method. Request MUST use the Cache-Control response directive "no-cache".

All LoST Sync responses, including those indicating a LoST warning or error, are carried in 2xx responses, typically 200 (OK). 3xx, 4xx and 5xx HTTP response codes indicates that the request itself failed or was redirected; these responses do not contain any LoST Sync XML elements.

6. RelaxNG

Note: In order to avoid copying pattern definitions from the LoST Relax NG schema [[RFC5222](#)] to this document we include it as "lost.rng" (XML syntax) in the Relax NG schema below.

```
<?xml version="1.0" encoding="utf-8"?>

  <grammar ns="urn:ietf:params:xml:ns:lostsync1"
    xmlns="http://relaxng.org/ns/structure/1.0"
    xmlns:a="http://relaxng.org/ns/compatibility/annotations/1.0"
    datatypeLibrary="http://www.w3.org/2001/XMLSchema-datatypes">

    <include href="lost.rng"/>

    <start combine="choice">

      <a:documentation> Location-to-Service Translation (LoST)
        Synchronization Protocol</a:documentation>

      <choice>
        <ref name="pushMappings"/>
        <ref name="pushMappingsResponse"/>
        <ref name="getMappingsRequest"/>
        <ref name="getMappingsResponse"/>
      </choice>
    </start>

    <define name="pushMappings">
      <element name="pushMappings">
        <oneOrMore>
          <ref name="mapping"/>
        </oneOrMore>

        <ref name="extensionPoint"/>
      </element>
    </define>

    <define name="pushMappingsResponse">
      <element name="pushMappingsResponse">
        <ref name="extensionPoint"/>
      </element>
    </define>

    <define name="getMappingsRequest">
      <element name="getMappingsRequest">
```



```
        <choice>
          <ref name="exists"></ref>
          <ref name="extensionPoint"/>
        </choice>
      </element>
    </define>

    <define name="exists">
      <element name="exists">
        <oneOrMore>
          <element name="mapping-fingerprint">
            <attribute name="source">
              <data type="token"/>
            </attribute>
            <attribute name="sourceId">
              <data type="token"/>
            </attribute>
            <attribute name="lastUpdated">
              <data type="dateTime"/>
            </attribute>
            <ref name="extensionPoint"/>
          </element>
        </oneOrMore>
      </element>
    </define>

    <define name="getMappingsResponse">
      <element name="getMappingsResponse">
        <oneOrMore>
          <ref name="mapping"/>
        </oneOrMore>
        <ref name="extensionPoint"/>
      </element>
    </define>

    <!-- error messages -->

    <define name="notDeleted">
      <element name="notDeleted">
        <ref name="basicException"/>
        <oneOrMore>
          <ref name="mapping"/>
        </oneOrMore>
      </element>
    </define>
  </grammar>
```


7. Operational Considerations

When different LoST servers use the mechanism described in this document to synchronize their mapping data then it is important to ensure that loops are avoided. The example shown in Figure 13 with three LoST servers A, B and C (each of them acts as a sync source and a sync destination) illustrates the challenge in more detail. A and B synchronize data between each other; the same is true for A and C, and B and C, respectively.

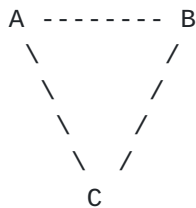


Figure 13: Synchronization Configuration Example

Now, imagine that server A adds a new mapping. This mapping is uniquely identified by the combination of "source", "sourceid" and "last updated". Assume that A would push this new mapping to B and C. When B obtained this new mapping it would find out that it has to distribute it to its peer C. C would also want to distribute the mapping to B. If the original mapping with the "source", "sourceid" and "last updated" is not modified by either B or C then these two servers would recognize that they already possess the mapping and can ignore the update.

It is important that implementations MUST NOT modify mappings they receive. An entity acting maliciously would, however, intentionally modify mappings or inject bogus mappings. To avoid the possibility of an untrustworthy member claiming a coverage region that it is not authorized for, authoritative mapping server MUST sign mappings they distribute using an XML digital signature [[W3C.REC-xmlsig-core-20020212](#)]. A recipient MUST verify that the signing entity is indeed authorized to speak for that region. In many cases, this will require an out-of-band agreement to be in place to agree on specific entities to take on this role. Determining who can speak for a particular region is inherently difficult unless there is a small set of authorizing entities that participants in the mapping architecture can trust. Receiving systems should be particularly suspicious if an existing coverage region is replaced by a new one that contains a different value in the <uri> element. With digitally signed mappings mappings cannot be modified by intermediate LoST servers.

8. Security Considerations

This document defines a protocol for exchange of mapping information between two entities. Hence, the protocol operations described in this document require authentication of neighboring nodes.

The LoST Sync client and servers MUST implement TLS and use TLS. Which version(s) ought to be implemented will vary over time, and depend on the widespread deployment and known security vulnerabilities at the time of implementation. At the time of this writing, TLS version 1.2 [[RFC5246](#)] is the most recent version, but has very limited actual deployment, and might not be readily available in implementation toolkits. TLS version 1.0 [[RFC2246](#)] is the most widely deployed version, and will give the broadest interoperability.

An additional threat is caused by compromised or misconfigured LoST servers. A denial of service could be the consequence of an injected mapping. If the mapping data contains an URL that does not exist then emergency services for the indicated area are not reachable. If all mapping data contains URLs that point to a single PSAP (rather than a large number of PSAPs) then this PSAP is likely to experience overload conditions. If the mapping data contains a URL that points to a server controlled by the adversary itself then it might impersonate PSAPs.

[Section 7](#) discusses this security threat and mandates signed mappings. For unusual changes to the mapping database approval by a system administrator of the emergency services infrastructure (or a similar expert) may be required before any mappings are installed.

9. IANA Considerations

9.1. Content-type registration for 'application/lostsync+xml'

This specification requests the registration of a new MIME type according to the procedures of [RFC 4288](#) [[RFC4288](#)] and guidelines in [RFC 3023](#) [[RFC3023](#)].

Type name: application

Subtype name: lostsync+xml

Required parameters: none

Optional parameters: charset

Indicates the character encoding of enclosed XML.

Encoding considerations: Identical to those of "application/xml" as described in [\[RFC3023\]](#), [Section 3.2](#).

Security considerations: This content type is designed to carry LoST Synchronization protocol payloads and the security considerations section of RFCXXXX is applicable. In addition, as this media type uses the "+xml" convention, it shares the same security considerations as described in [\[RFC3023\]](#), [Section 10](#). [NOTE TO IANA/RFC-EDITOR: Please replace XXXX with the RFC number of this specification.]

Interoperability considerations: None

Published specification: RFCXXXX [NOTE TO IANA/RFC-EDITOR: Please replace XXXX with the RFC number of this specification.]

Applications which use this media type: Emergency and Location-based Systems

Additional information:

Magic number(s): None

File extension(s): .lostsyncxml

Macintosh file type code(s): 'TEXT'

Person & email address to contact for further information: Hannes
Tschofenig <Hannes.Tschofenig@gmx.net>

Intended usage: LIMITED USE

Restrictions on usage: None

Author: Hannes Tschofenig <Hannes.Tschofenig@gmx.net>

Change controller:

This specification is a work item of the IETF ECRIT working group,
with mailing list address <ecrit@ietf.org>.

Change controller:

The IESG <iesg@ietf.org>

9.2. LoST Sync Relax NG Schema Registration

URI: urn:ietf:params:xml:schema:lostsync1

Registrant Contact: IETF ECRIT Working Group, Hannes Tschofenig
(Hannes.Tschofenig@gmx.net).

Relax NG Schema: The Relax NG schema to be registered is contained
in [Section 6](#).

9.3. LoST Synchronization Namespace Registration

URI: urn:ietf:params:xml:ns:lostsync1

Registrant Contact: IETF ECRIT Working Group, Hannes Tschofenig
(Hannes.Tschofenig@gmx.net).

XML:

BEGIN

```
<?xml version="1.0"?>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML Basic 1.0//EN"
  "http://www.w3.org/TR/xhtml1-basic/xhtml1-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 Synchronization Namespace</title>
</head>
<body>
  <h1>Namespace for LoST server synchronization</h1>
  <h2>urn:ietf:params:xml:ns:lostsync1</h2>
  <p>See <a href="[URL of published RFC]">RFCXXXX
    [NOTE TO IANA/RFC-EDITOR:
      Please replace XXXX with the RFC number of this
      specification.]</a>.</p>
</body>
</html>
END
```


10. Acknowledgments

Robins George, Cullen Jennings, Karl Heinz Wolf, Richard Barnes, Mayutan Arumaithurai, Alexander Mayrhofer, and Andrew Newton provided helpful input. Jari Urpalainen assisted with the Relax NG schema. We would also like to thank our document shepherd Roger Marshall for his help with the document.

We would like to particularly thank Andrew Newton for his timely and valuable review of the XML-related content.

We would like to thank Robert Sparks for his AD review feedback, Bjoern Hoehrmann for his media type review, and Julian Reschke and Martin Duerst for their applications area reviews.

11. References

11.1. Normative References

- [RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", [BCP 14](#), [RFC 2119](#), March 1997.
- [RFC2246] Dierks, T. and C. Allen, "The TLS Protocol Version 1.0", [RFC 2246](#), January 1999.
- [RFC2616] Fielding, R., Gettys, J., Mogul, J., Frystyk, H., Masinter, L., Leach, P., and T. Berners-Lee, "Hypertext Transfer Protocol -- HTTP/1.1", [RFC 2616](#), June 1999.
- [RFC2617] Franks, J., Hallam-Baker, P., Hostetler, J., Lawrence, S., Leach, P., Luotonen, A., and L. Stewart, "HTTP Authentication: Basic and Digest Access Authentication", [RFC 2617](#), June 1999.
- [RFC2818] Rescorla, E., "HTTP Over TLS", [RFC 2818](#), May 2000.
- [RFC3023] Murata, M., St. Laurent, S., and D. Kohn, "XML Media Types", [RFC 3023](#), January 2001.
- [RFC4288] Freed, N. and J. Klensin, "Media Type Specifications and Registration Procedures", [BCP 13](#), [RFC 4288](#), December 2005.
- [RFC5222] Hardie, T., Newton, A., Schulzrinne, H., and H. Tschofenig, "LoST: A Location-to-Service Translation Protocol", [RFC 5222](#), August 2008.
- [RFC5246] Dierks, T. and E. Rescorla, "The Transport Layer Security (TLS) Protocol Version 1.2", [RFC 5246](#), August 2008.
- [W3C.REC-xmlsig-core-20020212] Eastlake, D., Reagle, J., Solo, D., Hirsch, F., and T. Roessler, "XML-Signature Syntax and Processing", World Wide Web Consortium Second Edition REC-xmlsig-core-20020212, June 2008.

11.2. Informative References

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

Authors' Addresses

Henning Schulzrinne
Columbia University
Department of Computer Science
450 Computer Science Building
New York, NY 10027
US

Phone: +1 212 939 7004
Email: hgs+ecrit@cs.columbia.edu
URI: <http://www.cs.columbia.edu>

Hannes Tschofenig
Nokia Siemens Networks
Linnoitustie 6
Espoo 02600
Finland

Phone: +358 (50) 4871445
Email: Hannes.Tschofenig@gmx.net
URI: <http://www.tschofenig.priv.at>

