LIS description and standards compliance

Supported by.

NEMERGENT Initiative

Universidad del País Vasco · Euskal Herriko Unibertsitatea · CAMPUS OF INTERNATIONAL EXCELLENCE

Table of Contents

1 Scope 5
2 References 6
2.1 Normative References 6
3 Description 7
3.1 Supported Capabilities 7
4 Nemergent LIS Server/demo client 9
4.1 LIS Server 9
4.1.1 Overview 9
4.1.2 Installation using Docker (for the impatients) 9
4.1.3 Manual Installation 10
4.2 Publishing location 11
4.3 Nemergent LIS demo Client 11
4.3.1 Overview 11
4.3.2 Installation 11
4.3.3 Usage 12
5 Operation Samples 15
5.1 Location by value (HELD) 15
5.1.1 Diagram 15
5.1.2 Sample 15
5.2 Location by value OBO 16
5.2.1 Location by value (Using Device Identity Extension) 16
5.2.1.1 Diagram 16
5.2.1.2 Sample 17
5.3 Location by reference 17
5.3.1 Diagram 18
5.3.2 Sample 18
5.4 Publishing location with SIP 19
6 Validating 20
7 Changelog and contributors 21

List of Figures

Figure 1 NG112 architecture and LIS..... 5
Figure 2 Nemergent LIS Server structure and relationship with Kamailio..... 9
Figure 3 Location by value diagram (empty "locationRequest" element)..... 15
Figure 4 Location by value using Device Identity by IP and LI type filters..... 16
Figure 5 Location De-Reference request..... 18
Figure 6 Publish-ing location using SIMPLE Presence and retrieving later..... 20

List of Tables

Table 1 Analysis of LIS capabilities 7
Table 2 LIS Client options description 12
Table 3 Contributor table 21

1 Scope

This Document describes the NG112 (and NG911) LIS implementation contributed to the Nemergent initiative, together with a detailed analysis of its standard compliance.

A Location Information Server (LIS) as defined in [n1] supplies location, in the form of a PIDF-LO (location by value) or a location URI (location by reference). The LIS also provides a "dereference" service for a location URI it supplies: given the URI, the LIS provides the location value as a PIDF-LO. A LIS may be a database, or may be a protocol interworking function to an access network specific protocol.

Therefore, LIS comprises a LTD compliant interface for retrieving location information and some network specific protocol to gather such information for the underlying network location mechanisms (if any) or/and the endpoint itself. As a result, almost every element in the NG112/NG911 architecture may interact with the LIS (see Figure 1).

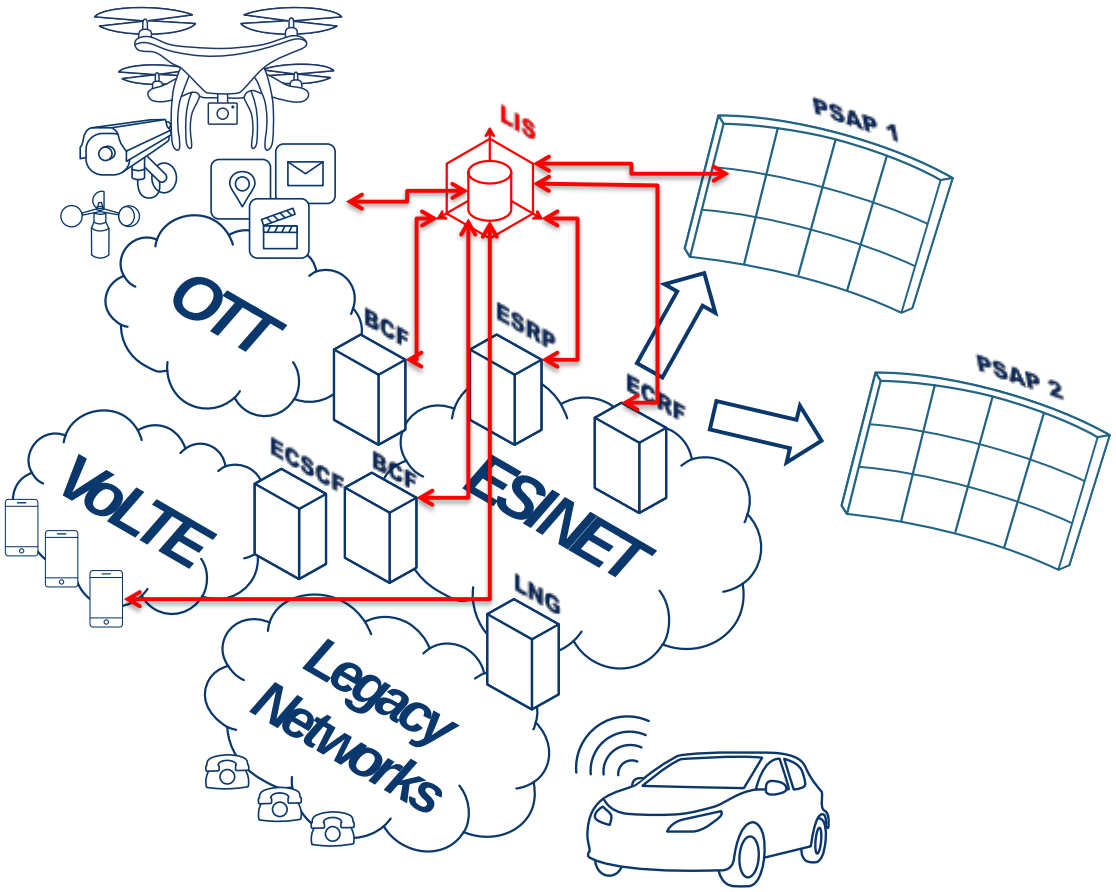

Figure 1 NG112 architecture and LIS

According to [n1] any LIS supplying location by reference must support HELD [n3] and/or SIP Presence Event Package [n7]. Additionally, LISs supporting SIP must support location filters [n8] and event rate control [n9].

Moreover, LISs queried by Legacy Network Gateways during the processing of:

- wireline emergency calls would typically use HELD with the identity extension [n5] using a telephone number as the identity and supply location by value in return.
- wireless emergency calls are usually protocol interwork functions between SIP or HELD and the legacy network's location determination subsystem. Typically they would supply location by reference.

2 References

2.1 Normative References

n1. EENA. Next Generation 112 Long Term Definition, Version 1.1, March 2013. http://www.eena.org/uploads/gallery/files/pdf/2013-03-15-eena_ng_longtermdefinitionupdated.pdf.

n2. Winterbottom, J., Thomson, M. and Tschofenig, H. GEOPRIV Presence Information Data Format Location Object (PIDF-LO) Usage Clarification, Considerations, and Recommendations, March 2009. RFC 5491, Internet Engineering Task Force.

n3. Barnes, M. HTTP-Enabled Location Delivery (HELD), September 2010. RFC 5985, Internet Engineering Task Force.

n4. IANA Geopriv http Enabled Location Delivery (HELD) Parameters Registry. http://www.iana.org/assignments/held-parameters/held-parameters.xhtml

n5. Winterbottom, J., Tschofenig, H. and Barnes, R. Use of Device Identity in HTTP-Enabled Location Delivery (HELD), March 2011. RFC 6155, Internet Engineering Task Force.

n6. Winterbottom, J., Tschofenig, H., Schulzrinne, H. and Thomson, M. A Location Dereference Protocol Using HTTP-Enabled Location Delivery (HELD), October 2012. RFC 6753, Internet Engineering Task Force.

n7. Rosenberg, J. A Presence Event Package for the Session Initiation Protocol (SIP), August 2004. RFC 3856, Internet Engineering Task Force.

n8. Mahy, R., Rosen, B., Tschofenig, H. Filtering Location Notifications in the Session Initiation Protocol (SIP), January 2012. RFC 6447, Internet Engineering Task Force.

n9. Niemi, A., Kiss, K., Loreto, S. Session Initiation Protocol (SIP) Event Notification Extension for Notification Rate Control, December 2011. RFC 6446, Internet Engineering Task Force.

n10. Bellis, R. Flow Identity Extension for HTTP-Enabled Location Delivery (HELD), April 2013. RFC 6915, Internet Engineering Task Force.

n11. Thomson, M. and Winterbottom, J. Revised Civic Location Format for Presence Information Data For- mat Location Object (PIDF-LO), February 2008. RFC 5139, Internet Engineering Task Force.

n12. J. Peterson, A Presence-Based GEOPRIV Location Object Format. RFC 4119, December 2005. Internet Engineering Task Force.

3 Description

3.1 Supported Capabilities

Table 1 Analysis of LIS capabilities

Capability	Reference	Protocol and specific features	Associated format/ XML Schema/ sub-namespace	Support in Nemergent LIS v1.0
HELD				
	RFC5985 [n3]	HELD basic definition	urn:ietf:params:xml:ns:geopriv:held Definition on Section 7 in RFC5985 [n3] MIME type: application/held+xml	YES
	RFC6155 [n5]	Support in HELD to retrieve location information on behalf of (OBO) other device, generally the endpoint, using some identification method (i.e. IP, URI, etc).	urn:ietf:params:xml:ns:geopriv:held:id Definition on Section 6 in RFC6155 [n5]	YES using sip: or tel: uri as id
	IANA Geopriv [n4]	Error codes defined by IANA in HELD for Geopriv location.	http://www.iana.org/assignments/held-parameters/held-parameters.xhtml In case of HTTP such errors will be carried as a 200 OK HTTP/HTTPS response	YES Supported error codes: requestError xmlError locationUnknown unsupportedMessage cannotProvideLiType notLocatable
	RFC6753 [n6]	Dereferencing mechanism in HELD.	Semantics as in RFC5985 [n3] Content-Type: application/held+xml Different Authorization schemes	YES Authorization by possesion
	RFC6915 [n10]	Flow-based identification mechanism in HELD.	urn:ietf:params:xml:ns:geopriv:held:flow	YES
SIP				
	RFC3856 [n7]	Presence Event Package for SIP (i.e. SUBSCRIBE/NOTIFY procedures).	MIME type: application/pidf-xml	YES
	RFC6447 [n8]	Definition of filters to limit when a subscriber gets asynchronous notification.	urn:ietf:params:xml:ns:location-filter	NO
	RFC6442 [n9]	Headers in SUBSCRIBE message indicating	max-rate-param	NO

		rate limiting mechanisms to control NOTIFY frequency.	min-rate-param amin-rate-param	Note: Kamailio's header processing capabilities + ratelimit module could be used.
PIDF-LO				
	RFC4119 [n9]	Addendum of geopriv element to status.	urn:ietf:params:xml:ns:pidf:geopriv10	YES
	RFC5139 [n11]	Civic for PIDF-LO.	urn:ietf:params:xml:ns:pidf :geopriv10:civicAddr	YES

4 Nemergent LIS Server/demo client

4.1 LIS Server

4.1.1 Overview

Figure 2 depicts overall architecture of the Nemergent proof-of-concept LIS Server. As shown, it is built upon SIP/HTTP capable multi-purpose Kamailio server which will be in fact responsible for providing both interfaces and presence+location information. Kamilio builtin MySQL backend will be used for permanent data storage.

Kamailio presence module is commonly used for SIMPLE Presence based presence+location handling in SIP networks. Therefore, SIP Publish-ing mechanism will be in fact used to feed the location information to the related *presentity* table.

In order to ensure a quick multiplatform installation and deployment, the only additional functionality added to Kamailio is the LIS.pl script (triggered by Perl module) and related configuration files.

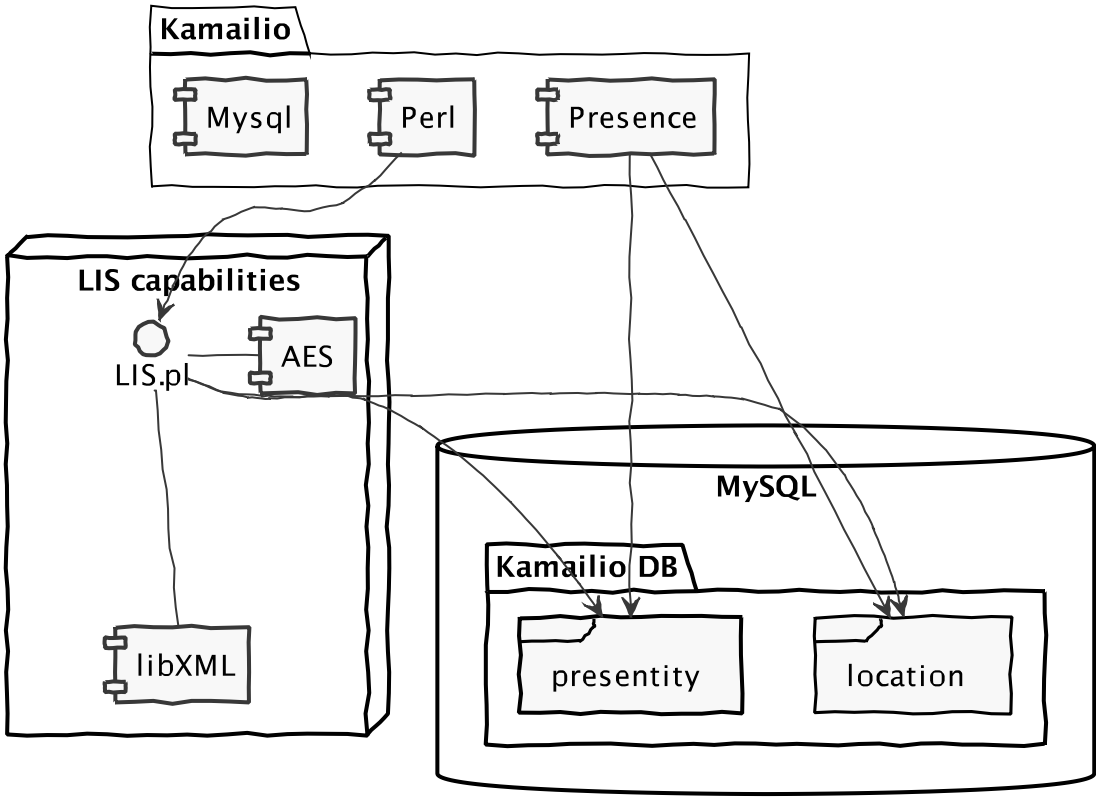

Figure 2 Nemergent LIS Server structure and relationship with Kamailio

4.1.2 Installation using Docker (for the impatients)

If the precompiled image is not available in the Docker Hub or if you prefer to build the image on your own, issue the following command at the root of the extracted LIS Server folder.

```
docker build -t nemergent/lis .
```

Caution: the process may take a long time and disk space

Run the created image

```
docker run -itd -p 81:81 -p 5060:5060/udp --name LIS -e
hostname=192.168.1.44 -e port=81  nemergent/lis
```

You need to set the hostname variable to the hostname/IP through which the LIS server can be accessed via HTTP, just the same with the port. If the suggested ports do not conform to your organization needs, you can change them, as below:

```
 docker run -itd -p 8081:81 -p 9060:5060/udp --name LIS -e
hostname=192.168.1.44 -e port=8081  nemergent/lis
```

This hostname/IP is reported on the Location Dereference URLs.

4.1.3 Manual Installation

Requirements:

- Kamailio SIP Server
- A MySQL Server (backend DB for Kamailio)
- Kamailio's Perl, Presence and MySQL modules
- Perl's CBC Crypt, AES and Try::Tiny modules

Procedure:

This howto will describe the steps to install the LIS server on Ubuntu 14.04 LTS 64 bits. Steps for the rest of the Ubuntu family should be similar, especially for the later versions.

Install Kamailio, MySQL server and all the dependencies

```
apt-get install \
kamailio \
kamailio-mysql-modules \
kamailio-presence-modules \
kamailio-perl-modules \
libcrypt-cbc-perl \
libdatetime-perl \
libtry-tiny-perl \
build-essential \
libssl-dev \
mysql-server \
autoconf \
libxml2-dev \
libxml-parser-perl
```

Install the AES, LibXML and XML::Tidy PERL modules from CPAN:

```
cpan Crypt::OpenSSL::AES
cpan XML::LibXML
cpan -f -i XML::Tidy
```

Edit the config file `/etc/default/kamailio` and uncomment the lines dealing with the start, user, and group configuration for kamailio:

```
# Set to yes to enable kamailio, once configured properly.
    RUN_KAMAILIO=yes
# User to run as
    USER=kamailio
# Group to run as
    GROUP=kamailio
```

Edit the config file `/etc/kamailio/kamctlrc` and uncomment the line dealing with the use of a MySQL engine:

```
DBENGINE=MYSQL
```

Create the following folder for kamailio to store run information:

```
mkdir -p /var/run/kamailio
```

Create the user for kamailio to run:

```
adduser --quiet --system --group --disabled-password \
--shell /bin/false --gecos "Kamailio" \
--home /var/run/kamailio kamailio
```

And give proper permissions:

```
chown kamailio:kamailio /var/run/kamailio
```

Copy the kamailio.cfg file to the default destination (`/etc/kamailio/kamailio.cfg`).

Copy the LIS.pl file to the Kamailio Perl folder `/usr/lib/x86_64-linux-gnu/kamailio/perl/`

Start MySQL (`service mysql start`) and run the `kamdbctl create` command and answer yes to all the questions.

To start kamailio just issue the following command:

```
kamctl start
```

4.2 Publishing location

Install sip-tester package (or build it from sources)

```
apt-get install sip-tester
```

Modify the PseudoVariables LOCALIP and LISADRESS in the provided `publish.sh` file.

Modify the `user-data.csv` file if you need to change some example data.

The format is `SIPURI;ALT;LONG;`

To execute the PUBLISH against the LIS Server, simply issue:

```
./publish.sh
```

4.3 Nemergent LIS demo Client

Nemergent LIS Software also packs a LIS demo Client which serves the purpose of testing the features and request types of the LIS Server.

4.3.1 Overview

The client is a lightweight implementation of several RFC protocols, which include HELD, Geopriv, HELD Device and Flow Identity extensions. The client implementation is not intended to serve as a reference implementation, but should suffice to show how to interact with the Nemergent LIS Server, and to provide a solid debugging platform for it.

The client is developed as a CLI application that must be run from a terminal, and accepts a series of options to define the desired behaviour.

4.3.2 Installation

Requirements:

- CMake > 2.6
- Curl
- LibXml2

This client is developed in pure C following the C89 standard, so it should be really portable to other architectures. By now it is tested on Linux x86, x86_64 and Linux armv7l.

To compile the client software a build script is provided in the make.sh file. In the vast majority of the systems it should suffice to execute:

```
./make.sh
```

And the LIS Client will be built in a new folder called `*build*`. To run the client, change to the `*build*` folder and refer to the usage patterns below or the sample runs described in the Samples section to use it.

4.3.3 Usage

```
LISClient --locReq --url=<HELDURL> [--types=<locTypes> --
exact --verbose]
```

```
LISClient --locReq --url=<HELDURL> [--types=<locTypes> --
exact --verbose] [--ip=<DEVICEIP> | --uri=<SIPURI> | --
flow=<FLOW>]
```

```
LISClient --locDeRef --url=<REFURL> [--types=<locTypes> --
method=<POST|GET> --verbose]
```

Table 2 LIS Client options description

Option	Explanation	Example(s)
--locReq	Indicates that a Location Request will be performed.	--locReq
--url	Sets the URL to be used for a Location Request or for a Location De-Reference.	--url http://lis.example.com:81/LIS/
--types	Sets the desired LI types to be requested to the server in a Location Request or Location De-Reference.	--types geodetic,civic,locationURI --types geodetic
--exact	Indicates that the server must supply ONLY the desired LI type or fail if it is not available. To use in a Location Request or Location De-Reference.	--exact
--ip	Sets the desired HELD Device Identity by specifying the device IP address in a Location Request	--ip 192.168.1.128
--uri	Sets the desired HELD Device Identity by specifying the device URI (sip or tel) in a Location Request	--uri sip:192.168.1.128 --uri tel:+34945123456
--flow	Sets the desired HELD Flow Identity by specifying the network flow that can be univocally associated with the device. This filter's syntax resembles a connection from the client to the server, setting first the client's IP and port, the server's same parameters, and finally the Network and Transport layer protocols. Syntax: IPO:PORTO-IPD:PORTD/l3/l4	--flow 192.168.1.128:4432-10.66.0.10:5060/ipv4/tcp
--locDeref	Indicates that a Location De-Reference will be performed.	--locDeref
--method	Sets the HTTP method to use when performing a Location De-Reference. It is compulsory to set this to POST when setting a –types filter in a Location De-Reference.	--method POST --method GET
--verbose	Indicates that the LIS Client should output more debug information, including the client request and the server's response. Very useful to	--verbose

	debug the protocols.	

5 Operation Samples

5.1 Location by value (HELD)

5.1.1 Diagram

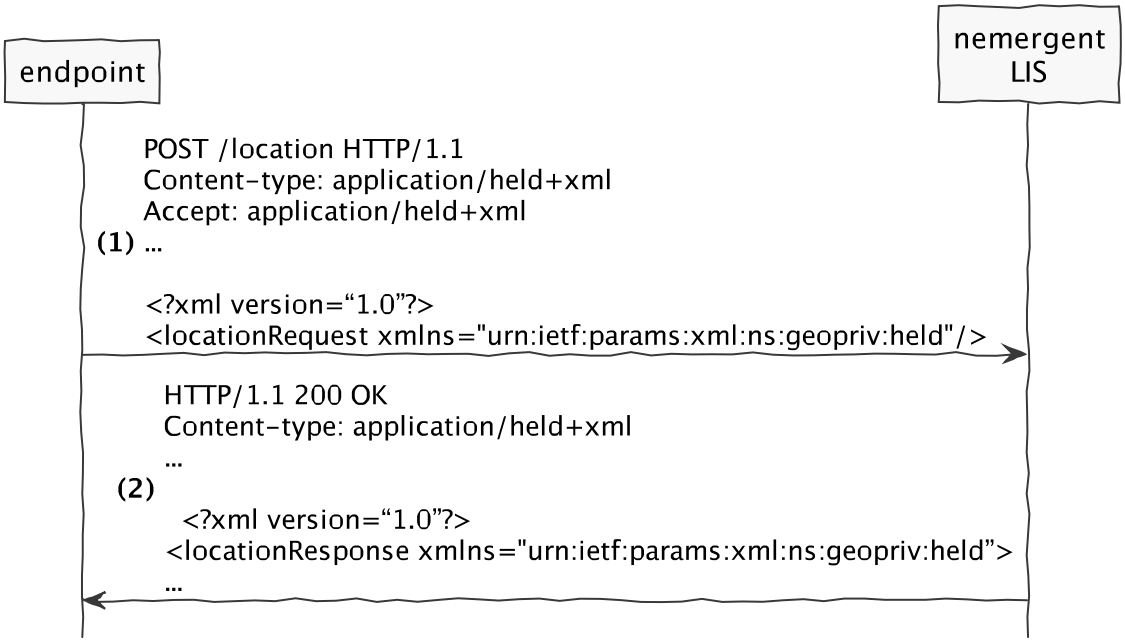

Figure 3 Location by value diagram (empty "locationRequest" element)

5.1.2 Sample

The most simple Location Request is the one in which the client requests its own LI (Location Information). In this case the "locationRequest" element is sent empty, and the server queries the database searching for the client's own IP address.

```
./LISClient --locReq --url http://10.66.0.60:81/LIS/
```

If a recent presence record is found, the LIS Client will print the following output:

```
[+] Location Request running...
[+] Server returned the following location information:
Presence Entity:
----------------
sip:client@192.168.1.128

Civic Location:
---------------
AU
NSW
Wollongong
North Wollongong
Flinders
Street
Campbell Street
Gilligan's Island
Corner
Video Rental Store
2500
Westerns and Classics
store
Private Box 15
```

```
Geodetic Location:
------------------
35.2225 -80.8449


Reference URL's:
----------------
http://10.66.0.60:81/LIS/locByRef/53616c7465645f5f7ceed110d12
74c8363d5c8ab6b42433d028d2d2b9bc66afd1c8ce285fc3686266d56a18b
602ea06a9973a7b1af8b2b24b4d7bc2fbc7301bf
```

For further packet inspection see the related pcap file (packet #4).

5.2 Location by value OBO

In more practical applications, the clients may not query for their own LI themselves, but other nodes in NG112/NG911 networks (ESINETs) would very likely need to retrieve, update, and check a 3rd party LI in order to perform emergency routing operations. In this case, the request is said to be performed **On Behalf of Others** or simply to be a **3rd Party Request**. In this case, the identity of the device to be queried can be specified by means of the **Device Identity Extension** or the **Flow Identity Extension** of the HELD Protocol.

5.2.1 Location by value (Using Device Identity Extension)

The Nemergent LIS Software supports the IP address and URI address use cases of the RFC 6155, and the Flow Identity Extension as described in RFC 6915.

5.2.1.1 Diagram

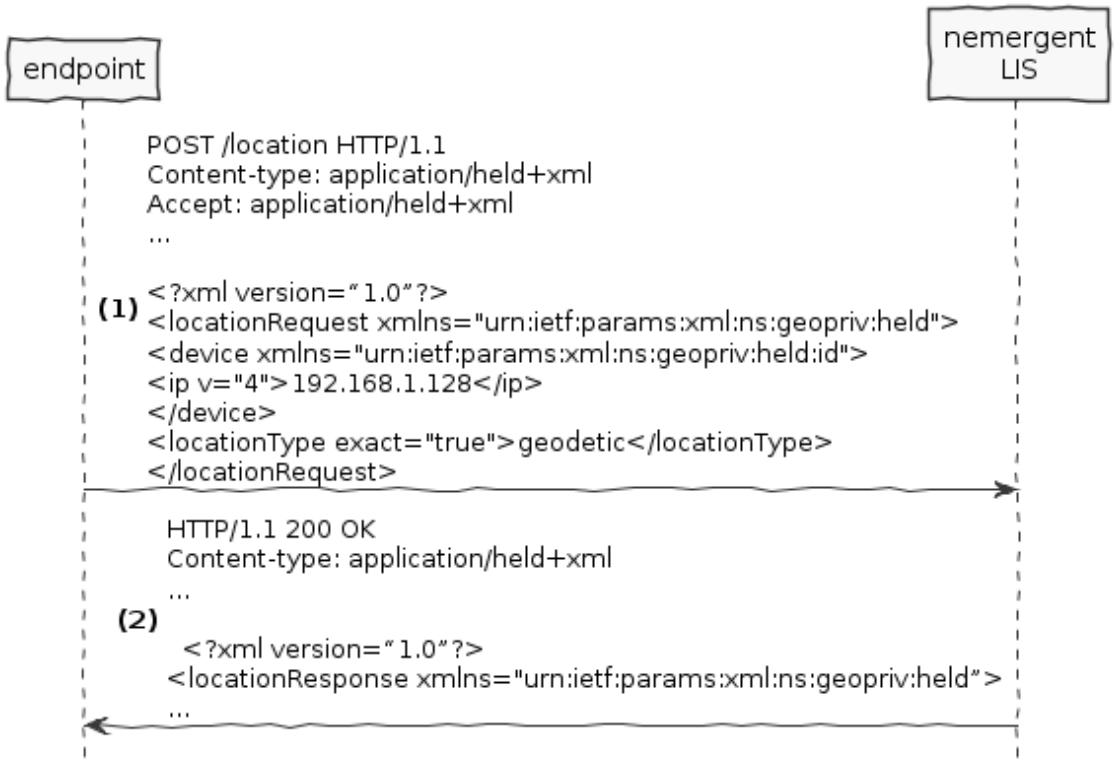

Figure 4 Location by value using Device Identity by IP and LI type filters

5.2.1.2 Sample

The LI of a 3rd party device can be queried by setting the `--ip`, `--uri` or `--flow` options of the LIS Client.

```
./LISClient --locReq --url http://10.66.0.60:81/LIS/ --ip
192.168.1.128 --types geodetic --exact
```

Here we are using the `--types geodetic` and the `--exact` options to illustrate how to request only one type of LI, geodetic in this case, and how to specify that ONLY that type of LI should be returned.

If the LIS database contains recent presence information about the device, the LIS Client will output the following information:

```
[+] Location Request running...
[+] Server returned the following location information:
Presence Entity:
----------------
sip:client@192.168.1.128

Civic Location:
---------------
(none)

Geodetic Location:
------------------
35.2225 -80.8449


Reference URL's:
----------------
(none)
```

Refer to the related pcap capture files for further inspection of the request (Packet #9).

5.3 Location by reference

Another requirement in NG112/NG911 networks, is to be able to get a reference to a LI piece, and to send it along the requests, in order to enable the upstream nodes to de-reference it and retrieve the actual Location Information. This is crucial, as the UE may update the LI regularly and those changes need to be taken into account as the call is handled within the network.

Location URI's produced by the Nemergent LIS Software are uniquely linked to an entity, release no private information until dereferenced, and have a configurable expiry time.

5.3.1 Diagram

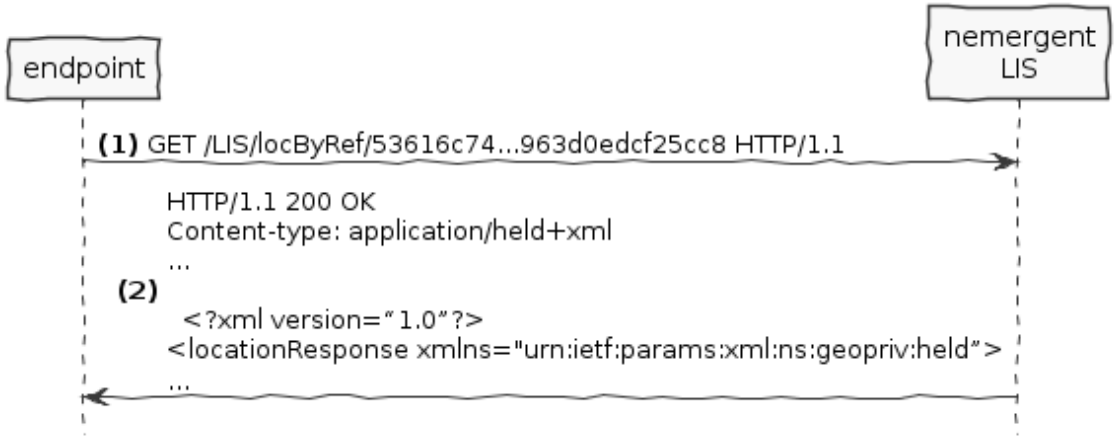

Figure 5 Location De-Reference request

5.3.2 Sample

First a Location Request has to be made, in order to obtain a Location URI (LI Reference). For example, we can request a single Location Reference using the `--types` and omitting the `--exact` options, as the LIS Server always returns a Location URI, if exact is not specified.

```
./LISClient --locReq --url http://10.66.0.60:81/LIS/ --ip
192.168.1.128 --types geodetic
```

The LIS Client will report the Location URI in the last section of the printed information.

```
[+] Location Request running...
[+] Server returned the following location information:
Presence Entity:
----------------
sip:client@192.168.1.128

Civic Location:
---------------
(none)

Geodetic Location:
------------------
35.2225 -80.8449


Reference URL's:
----------------
http://10.66.0.60:81/LIS/locByRef/53616c7465645f5f82182f29aff
8d83b9c120e6fb58e334f53a3b7889dd7e88d976530197f9cf6af94a78dcf
32663e9e24507610662cb3c107bb5985a656d424
```

Now we can perform the De-referencing operation by setting the `--locDeRef` and the `--url` options like below:

```
./LISClient --locDeRef --url
http://10.66.0.60:81/LIS/locByRef/53616c7465645f5fe9f20a84c24
1584de0106b0c66a07ae5a1ce92c8c91adb709af1ad9dbc01229efa5383bc
699c3557269e24bd725b6da48afd61361cf49138
```

The URL supplied will reference a previously selected Location Information, and, whenever it is retrieved it will supply the LIS Client with the most recent LI for that entity.

```
[+] Location DeReference running...
[+] Server returned the following location information:
Presence Entity:
----------------
sip:client@192.168.1.128

Civic Location:
---------------
AU
NSW
Wollongong
North Wollongong
Flinders
Street
Campbell Street
Gilligan's Island
Corner
Video Rental Store
2500
Westerns and Classics
store
Private Box 15


Geodetic Location:
------------------
35.2225 -80.8449


Reference URL's:
----------------
(none)
```

When a De-Reference is performed, the server will not return another Location URI.

5.4 Publishing location with SIP

The general workflow for which Nemergent LIS Software was designed is the one that is pictured below. In this scenario, UAs have the ability to determine location by themselves, or another equipment reports LI on behalf of them. In any case, LI is PUBLISH'ed through SIP to the LIS Server, which stores the LI received for the indicated SIP entity. Note that this method is just one among many others (i.e. supported by the network, SUPL, GMLC, etc…) to feed the LIS database.

Then, the received LI is made available to query through the HELD interface explained before.

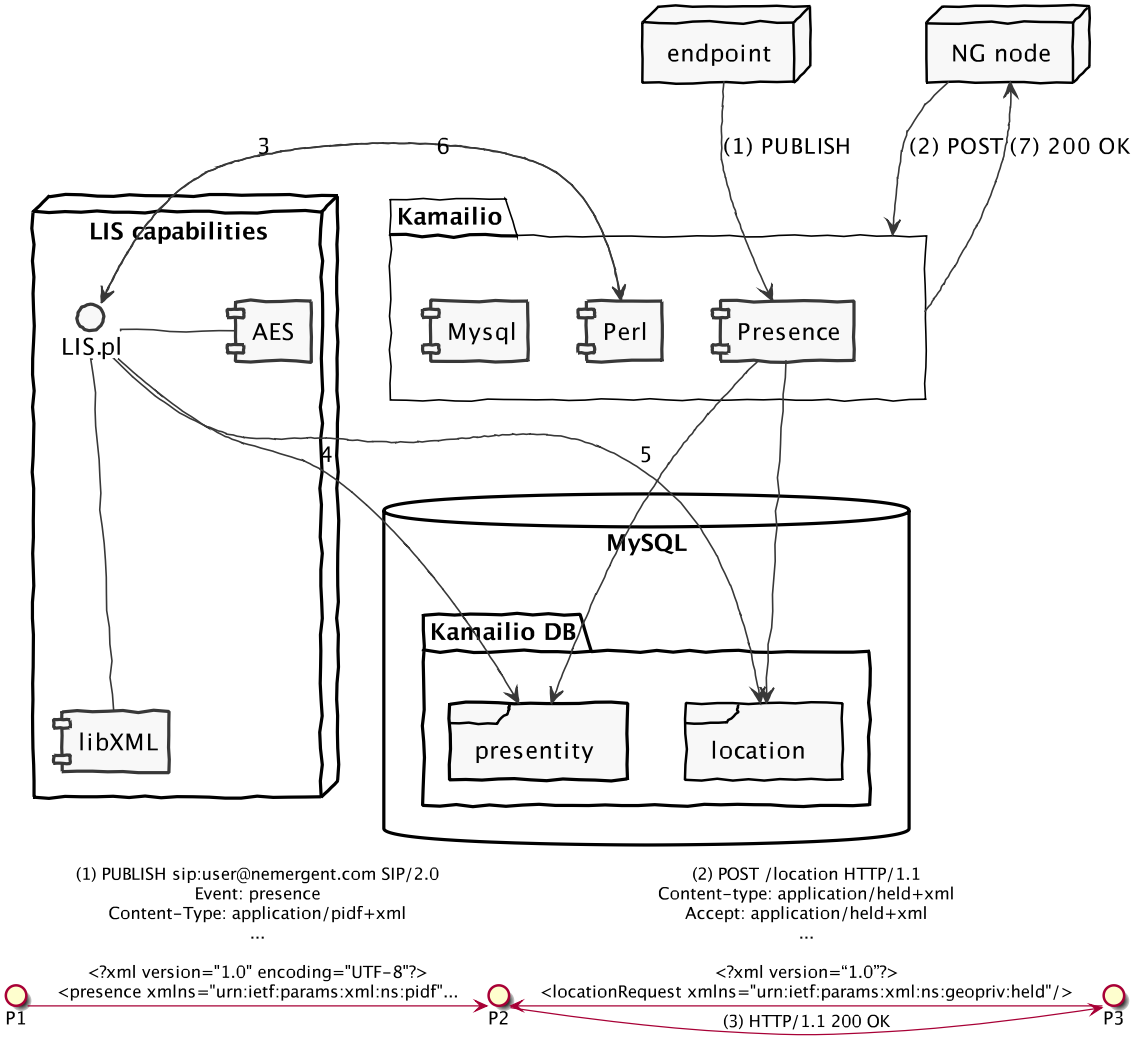

Figure 6 Publish-ing location using SIMPLE Presence and retrieving later

6 Validating

As XML is produced at the LIS Server to report the Location Information data, it must adhere to the proposed standards as described in the RFC's. Flattened XSD must be produced from the relevant IETF published XML schemas at IANA (http://www.iana.org/assignments/xml-registry/xml-registry.xhtml).

Any XML message produced by the Nemergent LIS Software is intended and extensively verified to conform to these schemas. These tests were conducted using the CLI tool named XMLint, whose usage is noted below.

```
xmllint --schema complete-lis-schema.xsd locationResponse-RFC5585.xml
```

7 Changelog and contributors

Table 3 Contributor table

Version	Contributors	Date	Changelog
1.0	Fidel Liberal fidel.liberal@ehu.eus Iñigo Ruiz-Relloso iruiz7@gmail.com	29 June 2016	Initial Version

Authors want specially to thank James Winterbottom for his comments and support.