ECRIT
Internet-Draft
Intended status: Informational
Expires: June 20, 2007

H. Schulzrinne
Columbia U.

December 17, 2006

Location-to-URL Mapping Architecture and Framework
draft-ietf-ecrit-mapping-arch-01

Status of this Memo

Copyright Notice

Abstract

This document describes an architecture for a global, scalable, resilient and administratively distributed system for mapping geographic location information to URLs, using the Location-to-Service (LoST) protocol. The architecture generalizes well-known approaches found in hierarchical lookup systems such as DNS.

Table of Contents

1. The Mapping Problem 3
2. Terminology 4
3. Definitions 4
4. Overview of Architecture 5
  4.1. Minimal System Architecture 6
5. Seeker 6
6. Resolver 7
7. Trees: Maintaining Authoritative Knowledge 8
  7.1. Basic Operation 8
  7.2. Answering Queries 10
  7.3. Overlapping Coverage Regions 10
  7.4. Scaling and Reliability 11
8. Forest Guides 11
9. Configuring Service Numbers 12
10. Security Considerations 14
11. IANA Considerations 15
12. Acknowledgments 15
13. References 15
  13.1. Normative References 15
  13.2. Informative References 15
Author's Address 16
Intellectual Property and Copyright Statements 17

1. The Mapping Problem

It is often desirable to allow users to access a service that provides a common function, but is actually offered by a variety of local service providers. In many of these cases, the service provider chosen depends on the location of the person wishing to access that service. Among the best-known public services of this kind is emergency calling, where emergency calls are routed to the most appropriate public safety answering point (PSAP), based on the caller's physical location. Other services, from food delivery to directory services and roadside assistance, also follow this general pattern. This is a mapping problem [8], where a geographic location and a service identifier (URN) [10] is translated into a set of URIs, the service URIs, that allow the Internet system to contact an appropriate network entity that provides the service.

The caller does not need to know where the service is being provided from, and the location of the service provider may change over time, e.g., to deal with temporary overloads, failures in the primary service provider location or long-term changes in system architecture. For emergency services, this problem is described in more detail in [6].

The overall emergency calling architecture [6] separates mapping from placing calls or otherwise invoking the service, so the same mechanism can be used to verify that a mapping exists ("address validation") or to obtain test service URIs.

Mapping locations to URIs describing services requires a distributed, scalable and highly resilient infrastructure. Authoritative knowledge about such mappings is distributed among a large number of autonomous entities that may have no direct knowledge of each other. In this document, we describe an architecture for such a global service. It allows significant freedom to combine and split functionality among actual servers and imposes few requirements as to who should operate particular services.

Besides determining the service URI, end systems also need to determine the local service numbers. As discussed in Section 9, the architecture described here can also address that problem.

The architecture described here uses the Location-to-Service Translation (LoST) [2] protocol, although much of the discussion would also apply for other mapping protocols satisfying the mapping requirements [8].

2. Terminology

In this document, the key words "MUST", "MUSTNOT", "REQUIRED", "SHALL", "SHALLNOT", "SHOULD", "SHOULDNOT", "RECOMMENDED", "MAY", and "OPTIONAL" are to be interpreted as described in RFC 2119 [1] and indicate requirement levels for compliant implementations.

3. Definitions

In addition to the terms defined in [8], this document uses the following terms to describe LoST clients and servers:

authoritative mapping server (AMS): An authoritative mapping server (AMS) is a LoST server that can provide the authoritative answer to a particular set of queries, e.g., covering a set of PIDF-LO civic labels or a particular region described by a geometric shape. In some (rare) cases of territorial disputes, two resolvers may be authoritative for the same region. An AMS may redirect or forward a query to other AMS within the tree.

child: A child is an AMS that is authoritative for a subregion of another AMS. A child can in turn be parent for another AMS.

(tree node) cluster: A node cluster is a group of LoST servers that all share the same mapping information and return the same results for queries. Clusters provide redundancy and share query load. Clusters are fully-meshed, i.e., they all exchange updates with each other.

forest guide: A forest guide has knowledge of the coverage region of all trees.

mapping: A mapping is a short-hand for 'mapping from a location object to one or more URLs describing either another mapping server or the desired service URLs'.

parent: A mapping server that covers the region of all of its children. A mapping server without a parent is a root resolver.

resolver: A resolver is contacted by a seeker, consults a forest mapping server and then resolves the query using an appropriate tree.

seeker: A seeker is a LoST client requesting a mapping. A seeker does not provide mapping services to others, but may cache results for its own use.

region map: A data object describing a contiguous area covered by an AMS, either as a subset of a civic address or a geometric object.

resolver: Resolvers contact authoritative mapping servers to answer queries by seekers, and may cache query results.

tree: A tree consists of a self-contained hierarchy of authoritative mapping servers. Each tree exports its coverage region to the forest mapping servers.

4. Overview of Architecture

In short, end users of the LoST-based [2] mapping mechanism, called seekers, contact resolvers that cache query results and know one or more "forest guides". Forest guides know the coverage region of trees and direct queries to the node at the top of the appropriate tree. Trees consist of authoritative mapping servers and maintain the authoritative mapping information. Figure 1 shows the interaction of the components.

```
       /-\        /-\        +-----+                 +-----+
      | S +******* R *********  FG *-----------------+  FG |
       \-/        \-/        |     |*                |     |
                             +--+--+  *              +--+--+
                                |      *                |
                                |       *               |
                                |        *              |
```

```
                                   |        *              |
                     /-\        +--+--+     *           +--+--+
                    | R +------>+  FG +-----*-----------+  FG |
                     \-/        |     |     *           |     |
                                +--+--+    *            +--+--+
                                   |      *                |
                                   |     *                 |
                                   |    *                  |
                                   |***                    ^
                                  / \                     / \
                                 /   \                   /   \
                                /     \                 /     \
                               /       \               /       \
                              -----------             -----------
                                tree                     tree
```

Architecture diagram, showing seekers (S), resolvers (R), forest guides (FG) and trees. The star (*) line indicates the flow of the query and responses in recursive mode.

Figure 1

The mapping function for the world is divided among trees. The collection of trees may not cover the whole world and trees are added and removed as the organization of mapping data changes. We call the collection of trees a forest. There is no limit on the number of

trees within the forest, but the author guesses that the number of trees will likely be somewhere between a few hundred and a few thousand. The lower estimate would apply if each country operates one tree. We assume that tree coverage information changes relatively slowly, on the order of less than one change per year per tree, although the system imposes no specific threshold. Tree coverage would change, for example, if a country is split or merged or if two trees for different regions become part of a larger tree. (On the other hand, information within a tree is likely to change much more frequently.)

4.1. Minimal System Architecture

It is possible to build a functioning system consisting only of seekers and resolvers if these resolvers have other means of obtaining mapping data. For example, a company acting as a mapping service provider could collect mapping records manually and make them available to their customers through the resolver. While feasible as a starting point, such an architecture is unlikely to scale globally. Among other problems, it becomes very hard for providers of authoritative data to ensure that all such providers have up-to-date information. If new trees are set up, they would somehow make themselves known to these providers. Such a mechanism would be similar to the old "hosts.txt" mechanism for distributing host information in the early Internet before DNS was developed.

Below, we describe the operation of each component in more detail.

5. Seeker

Clients desiring location-to-service mappings are known as seekers. Seekers are consumers of mapping data and originate LoST queries as LoST protocol clients. Seekers do not answer LoST queries. They contact either forest guides or resolvers to find the appropriate tree that can authoritatively answer their questions. Seekers can be end systems or call routing entities such as SIP proxy servers.

Seekers may need to obtain mapping information in several steps, i.e., they may obtain pointers to intermediate servers that lead them closer to the final mapping. Seekers MAY cache query results for later use, but otherwise have no obligations to other entities in the system.

Seekers need to be able to identify appropriate resolvers. The mechanism for providing seekers with that information is likely to differ depending on who operates the resolvers. For example, if the voice service provider operates the resolver, it might include the location of the resolver in the SIP configuration information it distributes to its user agents. An Internet access provider or enterprise can provide a pointer to a resolver via DHCP [5]. In an ad-hoc or zero-configuration environment, appropriate service directories may advertise resolvers.

Like other entities in the system, seekers can cache responses. This is particularly useful if the response describes the result for a civic or geospatial region, rather than just a point. For example, for mobile nodes, seekers would only have to update their resolution results when they leave the coverage area of a service provider, such as a PSAP for emergency services, and can avoid repeatedly polling for this information whenever the location information changes slightly. (Mobile nodes would also need a location update mechanism that is either local or triggered when they leave the current service area.) This will likely be of particular benefit for seekers representing a large user population, such as the outbound proxy in a corporate network. For example, rather than having to query separately for each cubicle, information provided by the authoritative node may indicate that the whole campus is covered by the same service provider.

Given this caching mechanism and cache lifetimes of several days, most mobile users traveling to and from work would only need to obtain service area information along their commute route once during each cache lifetime.

6. Resolver

A seeker can contact a forest guide (see below) directly, but may not be able to easily locate such a guide. In addition, seekers in the same geographic area may already have asked the same question. Thus, it makes sense to introduce another entity, known as a resolver in the architecture, that knows how to contact one or more forest guides and caches earlier queries to accelerate the response to mapping queries and to improve the resiliency of the system.

From a protocol perspective, a resolver acts in the same way as a seeker, except that it knows one or more forest guides.

ISPs or VSPs would include the address of a suitable resolver in their configuration information, i.e., in SIP configuration for a VSP or DHCP [5] for an ISP. Resolvers are manually configured with the name of one or more forest guides.

7. Trees: Maintaining Authoritative Knowledge

7.1. Basic Operation

The architecture assumes that authoritative knowledge about the mapping data is distributed among many independent administrative entities, but clients (seekers) needing the information may potentially need to find out mapping about any spot on earth. (Extensions to extra-terrestrial applications are left for future exploration.) Information is organized hierarchically, in a tree, with tree nodes representing larger geographic areas pointing to several child nodes each representing a smaller area. Each tree node can be a cluster of LoST servers that all contain the same information and back up each other.

Each tree can map a location described by civic and geographic coordinates for one type of service (such as 'sos.police', 'sos.fire' or 'counseling'), although nothing prevents re-using the same tree for multiple different services. The collection of all trees for one service is known as a forest.

Each tree node cluster knows the coverage region of its children and sends queries to the appropriate server "down" the tree. Each such tree node knows authoritatively about the service mappings for a particular region, typically, but not necessarily, contiguous. The region can be described by a polygon in geospatial coordinates or a set of civic address descriptors (e.g., "country = DE, A1 = Bavaria"). These coverage regions may be aligned with political boundaries, but that is not required. In most cases, to avoid confusion, only one cluster is responsible for a particular geographic or civic location, but the system can also deal with cases where coverage regions overlap.

There are no assumptions about the coverage region of a tree as a whole. For example, a tree could cover a single city, or a state/province or a whole country. Nodes within a tree need to loosely coordinate their operation, but they do not need to be operated by the same administrator.

The tree architecture is roughly similar to the domain name system (DNS), except that delegation is not by label, but rather by region. (Naturally, DNS does not have the notion of forest guides.) One can also draw analogies to LDAP, when deployed in a distributed fashion.

Tree nodes maintain two types of information, namely coverage regions and mappings. Coverage regions describe the region served by a child node in the tree and point to a child node for further resolution. Mappings contain an actual service URI leading to a service provider or another signaling server representing a group of service providers, which in turn might further route signaling requests to more servers covering smaller regions.

Leaf nodes, i.e., nodes without children, only maintain mappings, while tree nodes above the leaf nodes only maintain coverage regions. An example for emergency services of a leaf node entry is shown below, indicating how queries for three towns are directed to different PSAPs. Queries for Englewood are directed to another LoST server instead.

country	A1	A2	A3	resource
US	NJ	Bergen	Leonia	sip:psap@leonianj.gov
US	NJ	Bergen	Fort Lee	sip:emergency@fortleenj.org
US	NJ	Bergen	Teaneck	sip:police@teanecknjgov.org
US	NJ	Bergen	Englewood	lost:englewoodnj.gov

....

Coverage regions are described by sets of polygons enclosing contiguous geographic areas or by descriptors enumerating groups of civic locations. For the former, the LoST server performs a point-in-polygon operation to find the polygon that contains the query point. (More complicated geometric matching algorithms may be added in the future.)

For example, a state-level tree node for New Jersey in the United States may contain the following coverage region entries, indicating that any query matching a location in Bergen County, for example, would be redirected or forwarded to the node located at bergen.nj.example.org. There is no requirement that all child nodes cover the same level within the civic hierarchy. As an example, in the table below, the city of Newark has decided to be listed directly within the state node, rather than through the county. Longest-match rules allow partial coverage, so that for queries for all other towns within Essex county would be directed to the county node for further resolution.

C	A1	A2	A3	resource
US	NJ	Atlantic	*	lost:atlantic.nj.example.org/sos
US	NJ	Bergen	*	lost:bergen.nj.example.org/sos
US	NJ	Monmouth	*	lost:monmouth.nj.example.org/sos
US	NJ	Essex	*	lost:essex.nj.example.org/sos
US	NJ	Essex	Newark	lost:newark.example.com/sos

....

Thus, there is no substantial difference between coverage region and mapping data. The only difference is that coverage regions return LoST URLs, while mapping entries contain service URLs. Mapping

entries may be specific down to the house or floor level or may only contain street-level information. For example, in the United States, civic mapping data for emergency services is generally limited to address ranges ("MSAG data"), so initial mapping databases may only contain street-level information.

To automate the maintenance of trees, the LoST synchronization mechanism [11] allows nodes to query other nodes for mapping data and coverage regions. In the example above, the state-run node would query the county nodes and use the records returned to distribute incoming LoST queries to the county nodes. Conversely, nodes could also contact their parent nodes to tell them about their coverage region. There is some benefit of child nodes contacting their parents, as this allows changes in coverage region to propagate quickly up the tree.

7.2. Answering Queries

Within a tree, the basic operation is straightforward: A query reaches the root of the tree. That node determines which coverage region matches that request and forwards the request to the URL indicated in the coverage region record, returning a response to the querier when it in turns receives an answer (recursion). Alternatively, the node returns the URL of that child node to the querier (iteration). This process applies to each node, i.e., a node does not need to know whether the original query came from a parent node, a seeker, a forest guide or a resolver.

For efficiency, a node MAY return region information instead of a point answer. Thus, instead of returning that a particular geospatial coordinate maps to a service or mapping URL, it MAY return a polygon indicating the region for which this answer would be returned, along with expiration time (time-to-live) information. The querying node can then cache this information for future use.

For civic coordinates, trees may not include individual mapping records for each floor, house number or street. To avoid giving the wrong indication that a particular location has been found valid, LoST can indicate which parts of the location information have actually been used to look up a mapping.

7.3. Overlapping Coverage Regions

In some cases, coverage regions may overlap, either because there is a dispute as to who handles a particular geographic region or, more likely, since the resolution of the coverage map may not be sufficiently high. For example, a node may "shave some corners" off its polygon, so that its coverage region appears to overlap with its

geographic neighbor. For civic coordinates, houses on the same street may be served by different PSAPs. The mapping mechanism needs to work even if a coverage map is imprecise or if there are disputes about coverage.

The solution for overlapping coverage regions is relatively simple. If a query matches multiple coverage regions, the node returns all URLs, in redirection mode, or queries both children, if in recursive mode. If the overlapping coverage is caused by imprecise coverage maps, only one will return a result and the others will return an error indication. If the particular location is disputed territory, the response will contain all answers, leaving it to the querier to choose the preferred solution or trying to contact all services in turn.

7.4. Scaling and Reliability

Since they provide authoritative information, tree nodes need to be highly reliable. Thus, while this document refers to tree nodes as logical entities within the tree, an actual implementation would likely replicate node information across several servers, forming a cluster. Each such node would have the same information. Standard techniques such as DNS SRV records can be used to select one of the servers. Replication within the cluster can use any suitable protocol mechanism, but a standardized incremental update mechanism makes it easier to spread those nodes across multiple independently-administered locations. The techniques developed for meshed SLP [3] are applicable here.

8. Forest Guides

Unfortunately, just having trees covering various regions of the world is not sufficient as a client of the mapping protocol would not generally be able to keep track of all the trees in the forest. To facilitate orientation among the trees, we introduce a "forest guide". It is a server that keeps track of the coverage regions of the trees. For scalability and reliability, there will need to be a large number of forest guides, all providing the same information. A seeker can contact any forest guide and will then be directed to the right tree or, rarely, set of trees. Forest guides do not provide mapping information themselves.

Introducing forest guides avoids creating a global root, with the attendant management and control issues. Trees can also restrict their cooperation to parts of the information. For example, if country C does not recognize country T, C can propagate tree regions for all but T.

For authenticity, the records SHOULD be digitally signed. They are used by resolvers and possibly seekers to find the appropriate tree for a particular area. All forest guides should have consistent information, i.e., a collection of all the coverage regions of all the trees. A tree node at the top of a tree can contact any forest guide and inject new coverage region information into the system. One would expect that each tree announces its coverage to more than one forest guide. Each forest guide peers with one or more other guides and distributes new coverage region announcements to all other guides.

Forest guides fulfill a similar role to root servers in DNS. However, their number is likely to be larger, possibly counted in hundreds. They distribute information, signed for authenticity, offered by trees.

Forest guides can, in principle, be operated by anybody, including voice service providers, Internet access providers, dedicated services providers and enterprises.

As in routing, peering with other forest guides implies a certain amount of trust in the peer. Thus, peering is likely to require some negotiation between the administering parties concerned, rather than automatic configuration. The mechanism itself does not imply a particular policy as to who gets to advertise a particular coverage region.

9. Configuring Service Numbers

The section below is not directly related to the problem of determining service location, but is an instance of the more generic problem solved by this architecture, namely mapping location information to service-related parameters, such as service numbers.

For the foreseeable future, some user devices and software will emulate the user interface of a telephone, i.e., the only way to enter call address information is via a 12-button keypad with digits and the asterisk and hash symbol. These devices use service numbers to identify services. The best-known examples of service numbers are emergency numbers, such as 9-1-1 in North America and 1-1-2 in Europe. However, many other public and private service numbers have been defined, ranging in the United States from 3-1-1 for non-emergency local government services to 4-1-1 for directory assistance to various "800" numbers for anything from roadside assistance to legal services to home-delivery food.

Such service numbers are likely to be used until essentially all

---

communication devices feature IP connectivity and an alphanumeric keyboard. Unfortunately, for emergency services, more than 60 emergency numbers are in use throughout the world, with many of those numbers serving non-emergency purposes elsewhere, e.g., identifying repair or directory services. Countries also occasionally change their emergency numbers to conform to regional agreements. An example is the introduction of "1-1-2" for countries in Europe.

Thus, a system that allows devices to be used internationally to place calls needs to allow devices to discover service numbers automatically. In the Internet-based system proposed here [6], these numbers are strictly used as a human user interface mechanism and are generally not visible in call signaling messages, which carry the service URN [10] instead.

For the best user experience, systems should be able to discover two sets of service numbers, namely those used in the user's home country and in the country the user is currently visiting. The user is most likely to remember the former, but a companion borrowing a device in an emergency, say, may only know the local emergency numbers.

Determining home and local service numbers is a configuration problem, but unfortunately, existing configuration mechanisms are ill-suited for this purpose. For example, a DHCP server might be able to provide the local service numbers, but not the home numbers. When virtual private networks (VPNs) are used, even DHCP may provide numbers of uncertain origin, as a user may contact to the home network or some local branch office of the corporate network. Similarly, SIP configuration [4] would be able to provide the numbers valid at the location of the SIP service provider, but even a SIP service provider with national footprint may serve customers that are visiting any number of other countries.

Also, while initially there are likely to be only a few service numbers, e.g., for emergency services, the LoST architecture can be used to support other services, as noted. Configuring every local DHCP or SIP configuration server with that information is likely to be error-prone and tedious.

For these reasons, the LoST-based mapping architecture supports providing service numbers to end systems based on caller location. The mapping operation is almost exactly the same as for determining the service URL. The mapping can be obtained either along with determining the service URL or separately. The major difference between the two requests is that service numbers often have much larger regions of validity than the service URL itself. Also, the service number is likely to be valid longer than the service URL. Finally, an end system may want to look up the service number for its

home location, not just the current (visited) location.

10. Security Considerations

Security considerations for emergency services mapping are discussed in [9], while [10] discusses issues related to the service URN, one of the inputs into the mapping protocol. LoST-related security considerations are naturally discussed in the LoST [2] specification.

The architecture addresses the following security issues, usually through the underlying transport security associations:

Server impersonation: Seekers, resolvers, fellow tree guides and cluster members can assure themselves of the identity of the remote party by using the facilities in the underlying channel security mechanism, such as TLS.

Query or query result corruption: To avoid that an attacker can modify the query or its result, the architecture RECOMMENDS the use of channel security, such as TLS. Results SHOULD also be digitally signed, e.g., using XML digital signatures. Note, however, that simple origin assertion may not provide the end system with enough useful information as it has no good way of knowing that a particular signer is authorized to represent a particular geographic area. It might be necessary that certain well-known Certificate Authorities (CAs) vet sources of mapping information and provide special certificates for that purpose. In many cases, a seeker will have to trust its local resolver to vet information for trustworthiness; in turn, the resolver may rely on trusted forest guides to steer it to the correct information.

Region corruption: To avoid that a third party or an untrustworthy member of a server population introduces a region map that it is not authorized for, any node introducing a new region map MUST sign the object by encapsulating the data into a CMS wrapper. A recipient MUST verify, through a local policy mechanism, that the signing entity is indeed authorized to speak for that region. 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 region map is replaced with a new one with a new mapping address. In many cases, trust will be mediated: A seeker will have a trust relationship with a resolver. The resolver, in turn, will contact a trusted forest guide.

Additional threats that need to be addressed by operational measures include denial-of-service attacks [7].

11. IANA Considerations

Since this document describes an architecture, not a protocol, it does not ask IANA to register any protocol constants.

12. Acknowledgments

Richard Barnes, Jong Yul Kim, Andrew Newton, Murugaraj Shanmugam, Richard Stastny, and Hannes Tschofenig provided helpful comments.

13. References

13.1. Normative References

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

[2] Hardie, T., "LoST: A Location-to-Service Translation Protocol", draft-ietf-ecrit-lost-02 (work in progress), October 2006.

13.2. Informative References

[3] Zhao, W., Schulzrinne, H., and E. Guttman, "Mesh-enhanced Service Location Protocol (mSLP)", RFC 3528, April 2003.

[4] Petrie, D., "A Framework for Session Initiation Protocol User Agent Profile Delivery", draft-ietf-sipping-config-framework-09 (work in progress), October 2006.

[5] Schulzrinne, H., "A Dynamic Host Configuration Protocol (DHCP) based Location-to-Service Translation Protocol (LoST) Discovery Procedure", draft-ietf-ecrit-dhc-lost-discovery-00 (work in progress), December 2006.

[6] Rosen, B., "Framework for Emergency Calling in Internet Multimedia", draft-ietf-ecrit-framework-00 (work in progress), October 2006.

[7] Rosen, B. and J. Polk, "Best Current Practice for Communications Services in support of Emergency Calling", draft-ietf-ecrit-phonebcp-00 (work in progress), October 2006.

[8] Schulzrinne, H. and R. Marshall, "Requirements for Emergency Context Resolution with Internet Technologies", draft-ietf-ecrit-requirements-12 (work in progress), August 2006.

[9] Taylor, T., "Security Threats and Requirements for Emergency Call Marking and Mapping", draft-ietf-ecrit-security-threats-03 (work in progress), July 2006.

[10] Schulzrinne, H., "A Uniform Resource Name (URN) for Services", draft-ietf-ecrit-service-urn-05 (work in progress), August 2006.

[11] Schulzrinne, H., "Synchronizing Location-to-Service Translation (LoST) Servers", draft-schulzrinne-ecrit-lost-sync-00 (work in progress), November 2006.

Author's Address

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

Full Copyright Statement

Copyright (C) The Internet Society (2006).

This document is subject to the rights, licenses and restrictions contained in BCP 78, and except as set forth therein, the authors retain all their rights.

This document and the information contained herein are provided on an "AS IS" basis and THE CONTRIBUTOR, THE ORGANIZATION HE/SHE REPRESENTS OR IS SPONSORED BY (IF ANY), THE INTERNET SOCIETY AND THE INTERNET ENGINEERING TASK FORCE DISCLAIM ALL WARRANTIES, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO ANY WARRANTY THAT THE USE OF THE INFORMATION HEREIN WILL NOT INFRINGE ANY RIGHTS OR ANY IMPLIED WARRANTIES OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE.

Intellectual Property

The IETF takes no position regarding the validity or scope of any Intellectual Property Rights or other rights that might be claimed to pertain to the implementation or use of the technology described in this document or the extent to which any license under such rights might or might not be available; nor does it represent that it has made any independent effort to identify any such rights. Information on the procedures with respect to rights in RFC documents can be found in BCP 78 and BCP 79.

Copies of IPR disclosures made to the IETF Secretariat and any assurances of licenses to be made available, or the result of an attempt made to obtain a general license or permission for the use of such proprietary rights by implementers or users of this specification can be obtained from the IETF on-line IPR repository at http://www.ietf.org/ipr.

The IETF invites any interested party to bring to its attention any copyrights, patents or patent applications, or other proprietary rights that may cover technology that may be required to implement this standard. Please address the information to the IETF at ietf-ipr@ietf.org.

Acknowledgment

Funding for the RFC Editor function is provided by the IETF Administrative Support Activity (IASA).