Network Working Group
Request for Comments: 2969
Category: Informational

T. Eklof
L. Daigle

October 2000

Wide Area Directory Deployment - Experiences from TISDAG

Status of this Memo

This memo provides information for the Internet community. It does not specify an Internet standard of any kind. Distribution of this memo is unlimited.

Copyright Notice

Abstract

The TISDAG (Technical Infrastructure for Swedish Directory Access Gateway) project provided valuable insight into the current reality of deploying a wide-scale directory service. This document catalogues some of the experiences gained in developing the necessary infrastructure for a national (i.e., multi-organizational) directory service and pilot deployment of the service in an environment with off-the-shelf directory service products. A perspective on the project's relationship to other directory deployment projects is provided, along with some proposals for future extensions of the work (larger scale deployment, other application areas).

These are our own observations, based on work done and general project discussions. No doubt, other project participants have their own list of project experiences; we don't claim this document is exhaustive!

Table of Contents

1.0 Introduction

1.1 Overview of the TISDAG project

As described in more detail in [TISDAG], the original intention of the TISDAG project was to provide the infrastructure for a national whitepages directory service. To be effective, such an infrastructure needed to address the concrete realities of end-users' existing client software, as well as the needs of information providers ("Whitepages Directory Service Providers" -- WDSPs). These realities include the existence of multiple protocols (so-called directory service access protocols, as well as more general Internet application protocols such as HTTP and SMTP). The project was also sensitive to the fact that WDSPs have many good reasons for being reluctant to relinquish copies of their subscribers' personal data.

1.2 Organization of this document

In an effort to communicate the experiences with this project, from conception through implementation and pilot deployment, this document is divided into 3 major sections. The first section reviews specific lessons learned by the authors through the TISDAG project and implementation of one conformant system. Next, some perspectives are offered on the relationship of the TISDAG work to other large-scale directory projects that are currently on-going, to give a sense of how these efforts might possibly interact. Finally, some preliminary thoughts on applying the DAG system to other applications and deployment environments are outlined. Further suggestions for deploying networked DAG servers (meshes) can be found in [DAG-Mesh]. More discussion of useful development of architectural principles is provided in a separate document ([DAG++]).

2.0 The TISDAG project itself

2.1 TISDAG overview

Briefly, the technical infrastructure proposed for the TISDAG project (see [TISDAG] for the complete overview and technical specification) provides end-user client software with connection points to perform basic whitepages queries. Different connection points are provided for the various protocols end-users are likely to wish to use to access the information -- WWW (http), e-mail (SMTP), Whois++, LDAPv2 and LDAPv3. For each client, a transaction will be carried out within the bounds of the protocol's syntax and semantics. However, since the TISDAG system does not maintain a replicated copy of all whitepages information, but rather an index over the data that allows redirection (referrals) to services that are likely to contain responses that match the client's query, a fair bit of background work must be done by the DAG system in order to fulfill the client's query.

The first, and most important step, is for the system to make a query against the DAG Referral Index -- a server containing index information (obtained by the Common Indexing Protocol (see [CIP1, CIP2, CIP3]) in the Tagged Index Object format (see [TIO]). This index contains sufficient information to indicate which of the many participating WDSPs should be contacted to complete the query. Wherever possible, these referrals are passed back to the querying client so that it can contact relevant WDSPs directly. This minimizes the amount of work done by the DAG system itself, and allows WDSPs greater visibility (which is an incentive for participating in the system). Protocols which support referrals natively include Whois++ and LDAPv3 -- although these may only be referred to servers of the same protocol.

Since many protocols do not support referrals (e.g., LDAPv2), and in order to address referrals to servers using a protocol other than the calling client's own, a secondary step of "query chaining" is provided to pursue these extra referrals within the DAG system itself. For example, if an LDAPv2 client connects to the system, a query is made against the Referral Index to determine which WDSPs may have answers for the query, and then resources within the DAG system are used to pursue the query at the designated WDSPs' servers. The results from these different services are packaged into a single response set for the client that made the query.

The architecture that was developed in order to support the required functionality separated the system into distinct components to handle incoming queries from client software ("Client Access Points", or CAPs), a referral index (RI) to maintain an index over the collected whitepages information and provide referrals, based on actual data queries, to WDSPs that might have relevant information, and finally components that mediate access to WDSP whitepages servers to perform queries and retrieve results for the client's query ("Service Access Points", or SAPs). Several CAPs and SAPs exist within the system -- at least one for every protocol supported for incoming queries and WDSP servers, respectively.

Designed to be implementable as separate programs, these components interact with each other through the use of an internal protocol -- the DAG/IP. Pragmatically, the use of the protocol means that different components can reside on different machines, for reasons of load-balancing and performance enhancement. It also acts as a "common language" for the CAPs, SAPs and RI to express queries and receive results.

This outlines the planned or ideal behaviour of the system; once designed, a pilot phase was started for the project to compare reality against expectations. Two independent implementations of the software were created, and a test deployment was set up within the Swedish University Network (SUNET). More detail on the project and its current status can be found at http://tisdag.sunet.se/.

The rest of this section outlines some conclusions drawn from making a reality of the proposed architecture -- both successes and surprises.

2.2 Some successes

Implementation and pilot deployment of software meeting the TISDAG technical specification did demonstrate some important successes of the approach.

Most notably, the system works pretty much as expected (see exceptions below) to provide transparent middleware for whitepages directory services. That is, client software and WDSP servers were minimally affected -- from the point of view of behaviour and configuration, the DAG system looked like a server to clients, and a client to servers.

The goal of the TISDAG project, operationally, was to be able to provide responses to end-user queries in reasonable response times (although not "an addressbook replacement"). The prototype systems demonstrated some success in achieving responses within 10 seconds, at least with the limited testbed of a configuration with 10 WDSP's providing directory service information. More observations on system performance are provided below.

The DAG system does demonstrate that it is possible to build referral-level services at a national level (although the deployment has yet to prove conclusively that it can, in its current formulation, operate as a transparent query-fulfillment proxy service).

The success of the implementation demonstrated that it is possible, in some sense, to do (semantic) protocol mapping with N+M complexity instead of NxM mappings. That is, protocol translations had to be defined for "N" allowable end-user query access protocols to/from the DAG/IP, and "M" supported WDSP server protocols, instead of requiring each of the N input components to individually map to the M output protocols.

As a correlated issue, the prototype system demonstrated some successes with mapping between schema representations in the different protocol paradigms -- in a large part because system's schemas were kept simple and focused on the minimal needs to support the base service requirements.

2.3 Some surprises

Over the span of a dozen months from the first "final" document of the specification through the implementation and first deployment of the software system, a few surprises did surface. These fell into two categories: those that surfaced when the theoretical specification was put into practice, and others that became apparent when the resulting system was put into operation with commercial software clients and servers.

More detail is provided in the Appendix concerning specific software issues encountered, but some of the larger issues that surfaced during the implementation phase are describe below.

2.3.1 LDAP objectclasses and the "o" attribute

It came as a considerable surprise, some months into the project, that none of the "standard" LDAP person objectclasses included organization ("o") as an attribute. The basic assumption seems to be that "o" will be part of the distinguished name for an entry, and therefore there is little (if any) cause to list it out separately. This does make it trickier to store information for people across multiple organizations (e.g., at an ISP's directory server) and use the organization name in query refinement. (Roland Hedberg caught this issue, and has flagged it to the authors of the "inetorgperson" objectclass document).

2.3.1 The Tagged Index Object

The Tagged Index Object ("TIO"), used to carry indexes of WDSP information to the RI, is designed to have record (entry) tags to reduce the number of false positive referrals generated when doing a search in the RI. One of the features of the first index object type, Whois++'s centroid (see [centroid]) was the fact that the index object size did not grow linearly with the size of data indexed -- i.e., at some point the growth of the index object slowed as compared to that of the underlying data set. At first glance, this also seems to be the case for the TIO. However, as the index grows in size the compression factor of the TIO may not achieve the same efficiency as the centroids. One reason for this is that the tagged lists can get quite long, depending on the ordering of the assignment of tags to the underlying data. That is, the tagging as defined allows for a compressed expression of tag "ranges" -- e.g., "1-500" instead of "1,2,3,[...]500". Thus, it might be interesting to explore an optimal "sorting" of underlying data, before applying tags, in order to arrange the most common tokens have consecutive tags (maximal compression of the tag lists). It's not clear if this can be done efficiently over the entire set of records, attributes, and tokens, but it would bear some investigation, to produce the most compressed TIO for transmission.

Additionally, in order to make (time) efficient use of the tags in the RI in practice, it is almost necessary to "reinflate" the index object to be able to do joins on tag lists associated with tokens that match. Alternatively, the compressed tag list can be stored, and there is an additional cost associated with comparing the tag lists for matching tokens -- i.e., list comparison operations done outside the scope of a base database management system. There was an unexpected tradeoff to be made.

2.3.3 Handling Status Messages

Mapping of status messages from multiple sub-transactions into a single status communication for the end-user client software became something of a challenge. When chaining a query to multiple WDSPs (though the SAPs), it is not uncommon for at least one of the WDSP servers to return an error code or be unavailable. If one WDSP cannot be reached, out of several referrals, should the client software be given the impression that the query was completed successfully, or not? Most client protocol error handling models are not sophisticated enough to make this level of distinction clear.

2.3.4 Deployment with Commercial Software

When it then was time to test the resulting software with standard commercial client and server software, a few more surprises came to light (primarily in terms of these softwares' expected worldview and occasional implementation shortcuts). Again, more detail is provided in the Appendix, but highlights included client software that could only handle a very small subset of a protocol's defined status message lexicon (e.g., 2 system messages supported), and client software that automatically appended additional terms to a query specified by the user (e.g., adding "or email=<what the user typed in to the query>").

2.4 Some observations

2.4.1 Participation of the WDSPs

One of the things that came to light was that the nature of the index object generated by the WDSPs has an important impact on performance -- both in terms of integrating the index object into the Referral Index, and in terms of efficiency of handling queries. A proposal might be either to define more clearly how the WDSPs should generate the CIP index object (currently left to their discretion), or to alert individual WDSPs when their index objects are considered substandard.

On another front, when chaining referrals to WDSP servers, some servers perform more efficiently than others, affecting the overall response time of the DAG system. From a service point of view, it should also be possible to suggest to WDSP's that are consistently slow (longer than some selected response time) that they are substandard.

2.4.2 Index Objects and Referral Index size

As described in more detail [complex], there are many factors that can influence the growth factor of index objects (as more data is indexed). That work dealt specifically with tokenized data for Whois++ centroids, and is not immediately generalizable to all forms of the Tagged Index Object. However, the particular structure of the TIO used for the TISDAG project is similar enough in structure to a centroid that the same "order of magnitude" and growth characteristics are applicable.

Factors that affects the size of the data ("number of entries"):

- Number of generated tokens
  The number of tokens generated from the directory data depends on what is tokenized. If data is tokenized on names and addresses (i.e. not unique data like phone numbers) a rough estimation is that the number_of_tokens = 0.2 * number_of_data_records. The growth is linear in the span from a few thousand to at least 1.2 million records. The growth should then level off since the sets of names and addresses are finite, but the current tests have not shown a break point.

  If data is tokenized on something that is unique, e.g. phone numbers, then a rough estimation is that the number_of_tokens = number_of_data_records. Note that it is possible to tokenize in different ways, for example divide the phone numbers in parts. This would result in fewer tokens.

- Number of directories
  Since the tokens are generated individually for each directory, the data size depends on the number of directories. 10 directories with 100.000 records will generate the same amount of tokens as one directory with 1.000.000 records.

2.4.3 Index Object and Query Performance

Factors that affects the performance ("queries/second"):

- Type of query (exact, substring, etc.)
  A ’substring’ query is slower than an ’exact’ query due to:
  1) somewhat slower look-up in the internal DAG database than an exact query.
  2) Mostly, a larger amount of data is fetched from the internal DAG database due to more hits, which generates more index processing.

3) Substring queries are sent to the directory servers which also results in more hits and more data fetched. The directory servers may also be more or less effective in handling substring queries.

. Number of search attributes
A query with one or few attributes will most of the time result in many hits, which results in a lot of data, both internally in DAG and from the directory servers. On the other hand, a query with many attributes will result in a somewhat slower look-up in the internal DAG database.

. Number of directories
A larger number of directories may result in many referrals, but it depends on the query. A simple query will generate a lot of referrals, which means a lot of data from the directories has to be fetched. It will also result in a somewhat slower look-up in the internal DAG database.

. Number of chained referrals
Queries that are not chained are faster, since the result data does not have to be sent through the DAG system. Chained queries to several directories can be processed in parallel in the SAPs, but all data has to be processed in the CAP before sent to the client.

. Response time in the directory servers
The response time from the directory servers are of course critical. The total response time for DAG is never faster than the slowest involved directory server.

. Number of tokens (size of Tagged Index Objects)
The number of tokens has little impact on the look-up time in the internal DAG database.

2.5 Some evolutions

To date, the TISDAG project has been "alive" for just over two years. During that time, there have been a number of evolutions -- in terms of technologies and ideas outside the project (e.g., user and service provider expectations, deployment of related software, etc) as well as goals and understanding within the scope of the project.

Chief among these last is the fact that the project set out to primarily fulfill the role of a national referral service, and gradually evolved towards becoming more of a transparent protocol proxy service, fulfilling client queries as completely as possible, within the client protocol’s semantics. This evolution was probably

provoked by a number of reasons -- existing client & server software has a narrower range of accepted (expected) behaviour than their protocol specs may describe, once the technology was there for some proxying, going all the way seemed to be within reach, etc.

>From the point of view of providing a national whitepages service, this is a very positive evolution. However, it did place some strains on the original system architecture, for which some adjustments have been proposed (more detail below). What is less clear is the impact this evolution will have on the flexibility of the system architecture -- in terms of addressing other applications, different protocols (and protocol paradigms), etc. That is, the original intention of the system was to very simply fulfill an unsophisticated role -- "find things that sort of match the input query and let the client itself determine if the match is close enough". As the requirements become more sophisticated, the simplicity of the system is impacted, and perhaps more brittle. (Some proposals for avoiding this are outlined in [DAG++], which attempts to return to the underlying principles and propose steps forward at that level).

In terms of impact within the TISDAG project, this evolution lead to the following technical adjustments:

. The latest version of the technical specification makes a distinction (in the internal protocol grammar) between queries directed at the Referral Index, and those passed to SAPs to fulfill a query. This distinction keeps the query-routing queries simple, but allows more sophistication in expressing a query designed to fulfill the client’s original semantic expression.

. The additional constraints in the SAP query language is still not enough to allow the internal protocol to express very sophisticated queries. Originally intended only for query-routing queries, the DAG/IP expects all queries to be token-based (whereas LDAP queries are phrase-oriented). This means that SAPs have to do a good deal of "post-pruning" of WDSP result sets to match the DAG/IP query sent by a CAP for query fulfillment. And, CAPs must in turn do more post-pruning to match the DAG/IP results (from the SAPs) to the original query semantics.

The real strength of the TISDAG project was that it separated the technical framework needed to support the service from the configuration required in order to support a particular application or service -- query & schema mapping, configuration for protocols,

etc. Future improvements should focus on evolving that framework, maintaining the separation from the specific applications, services, and protocols that may use it.

3.0 Related Projects

The TISDAG project is not alone in attempting to solve the problems of providing coordinated access to resources managed by multiple, disparate services.

3.1 The Norwegian Directory of Directories (NDD)

Described in [NDD], the Norwegian Directory of Directories project also aims to provide necessary infrastructure for a national directory service. It assumes LDAP (v2 or v3) accessibility of WDSP information (provided by the WDSP itself, or through other arrangements), and aims to resolve some of the trickier issues associated with hooking together already-operational LDAP servers into a coherent network: uniform distinguished naming scheme, and content-based referrals. It also addresses some of the pragmatic realities of being compatible with different versions of LDAP clients -- e.g., v2, which does not support referrals, and v3, which does.

At the heart of the system is the "Referral Index and Organizational information" (RIO) server, which provides a searchable catalogue over Norwegian organization. This facilitates the location of whitepages servers for individual organizations (assuming the query includes information about which organization(s) is(are) interesting).

This work can be seen as being complementary to the TISDAG work, in that it provides a more focused service for integrating LDAP directory servers. However, there is still some requirement that one knows the organization to which a person belongs before doing a search for their e-mail address. This may be reasonable for seeking mail addresses associated with a person’s work organization, but is less often successful when it comes to finding a personal e-mail address -- in an age where ISPs abound, a priori knowledge of a user’s ISP identification is unlikely.

3.2 DESIRE Directory Services

The EC funded project DESIRE II (http://www.desire.org) is developing a distributed European indexing system for information on Research and Education. The Directory Services work undertaken by DANTE and SURFnet proposes an architecture applied to a server mesh structure to create a wide-area directory service infrastructure.

This service is intended to support both whitepages information with LDAP servers at WDSPs, as well as a Web-search meshes at various places using Whois++ for information about resources and routing of queries to other index-based services.

Like the TISDAG project, the DESIRE directory services project aims to act as a focal point for queries, allowing client software to access appropriate resources from a wide range of disparate services.

There are architectural differences between the approach used in the TISDAG project and the DESIRE directory service project, but many of the driving needs are the same, and the approach of using content-based indexing and referrals was also selected.

4.0 Some Directions for TISDAG Next Steps

The fun thing with technology is that there are always more tweaks and changes that can be made. However, a service should evolve in response to specific customer needs, and there are several ways in which the TISDAG service itself could advance. Some of them are outlined below, in terms of possibilities perceived at this time, rather than specific recommendations for underlying technology changes that would be necessary to fulfill them. A related topic, networking DAG servers (meshes), is discussed in [DAG-Mesh].

4.1 Security support

There is a need for security considerations when making use of a wide-scaled directory system in other application areas than the public white-pages application of the TISDAG project. There are issues whether the directory service is distributed across the Internet, or even if it functions completely within an internal, closed network.

4.2 WDSPs attributes and schemas

Today the DAG system makes use of 2 information schemas -- the DAGPERSON schema for information about specific people, and the DAGORGROLE schema for organizational roles. The technical specification includes a definition of the schema, as well as an understood mapping to (and from) some standard schemas used in the supported protocols. Nevertheless, to include new WDSPs which may not have all attributes in schemas, may use different schemas as well as query attributes, it should be possible to provide creation and use of new customized/standardized schemas and perform schema mapping if it’s necessary. It might also be possible to constrain queries to desired query attributes, templates, or object classes.

In practice, this means that different WDSP's may choose to use different subparts of one defined schema, or even implement local customizations.

5.0 Some conclusions

Although fewer people now hold out the hope of a unified global directory service, based on standardize protocols, it is interesting to see more projects providing infrastructure that permits unified access to what is otherwise an unforgivingly diverse and dislocated set of information servers. What cannot be dictated (in standardized protocols and schemas) may yet be accommodated through service infrastructure. The right approach seems to be to build better and better frameworks for supporting such diversified services, without making the framework architecture dependent on specific technologies.

6.0 Security Considerations

To date, the TISDAG project has focused on serving only publicly-sharable information. As noted in Section 4.1, any future work will have to provide additional facilities for providing authentication, authorization, encryption, and otherwise handling sensitive data in an open environment.

7.0 Acknowledgements

This document outlines the perspectives and opinions of the authors, based on experience as well as many fruitful and enlightening discussions with others: Roland Hedberg, Torbjorn Granat, Patrik Granholm, Rikard Wessblad and Sandro Mazzucato.

The work described in this document was carried out as part of an on-going project of Ericsson. For further information regarding that project, contact:

Bjorn Larsson
bjorn.x.larsson@era.ericsson.se

8.0 Authors' Addresses

Thommy Eklof
Hotsip AB

EMail: thommy.eklof@hotsip.com

Leslie L. Daigle
Thinking Cat Enterprises

EMail: leslie@thinkingcat.com

9.0 References

Request For Comments (RFC) and Internet Draft documents are available from numerous mirror sites.

[CIP1] Allen, J. and M. Mealling, "The Architecture of the Common Indexing Protocol (CIP)", RFC 2651, August 1999.

[CIP2] Allen, J. and M. Mealling, "MIME Object Definitions for the Common Indexing Protocol (CIP)", RFC 2652, August 1999.

[CIP3] Allen, J., Leach, P. and R. Hedberg, "CIP Transport Protocols", RFC 2653, August 1999.

[DAG++] Daigle, L. and T. Eklof, "An Architecture for Integrated Directory Services", RFC 2970, October 2000.

[DAG-Mesh] Daigle, L. and T. Eklof, "Networking Multiple DAG servers: Meshes", RFC 2968, October 2000.

[TISDAG] Daigle, L. and R. Hedberg "Technical Infrastructure for Swedish Directory Access Gateways (TISDAG)," RFC 2967, October 2000.

[centroid] Deutsch, P., Schoultz, R., Faltstrom, P. and C. Weider, "Architecture of the WHOIS++ service", RFC 1835, August 1995.

[NDD] Hedberg, R. and H. Alvestrand, "Technical Specification, The Norwegian Directory of Directories (NDD)", Work in Progress.

[TIO] Hedberg, R., Greenblatt, B., Moats, R. and M. Wahl, "A Tagged Index Object for use in the Common Indexing Protocol", RFC 2654, August 1999.

[complex] P. Panotzki, "Complexity of the Common Indexing Protocol: Predicting Search Times in Index Server Meshes", Master's Thesis, KTH, September 1996.

[WAP] The Wireless Application Protocol, http://www.wapforum.org

Appendix -- Specific Software Issues and Deployment Experiences

The following paragraphs outline practical deployment experiences in an anecdotal fashion. This is not meant to be construed as an exhaustive, authoritative evaluation of existing client software, but rather an indication of the types of challenges the average implementation team may expect to encounter in a development and deployment effort.

Character encoding

One client's addressbook sends iso-8859 encoding (depending on the font configuration in the browser) when querying a directory server but the directory server responds with Unicode (UTF-8) encoding. This means that the LDAP CAP would have to handle different character set encodings for request and response.

Referrals

Today there appears to be only one commercial addressbook supporting LDAPv3. All the others support only LDAPv2. However, this LDAPv3 client software does not handle referrals correctly -- the client couldn't handle server the result contains "response code 10" (designated for referrals). From what was observed, there was now way for the client or the end-user to decide if, or which, referrals to follow-up. It is therefore not clear how the LDAP clients handle a combination of both referrals and results -- but the supposition is that it doesn't work.

Objectclasses in LDAP

No objectclass is defined in the query to the DAG-system from the LDAP-clients. This means that the DAG-system doesn't see any differences between "inetOrgPerson" and "organisationalRole" when attribute "cn" is representing both "name" and "role". This is not so much a problem as that it has interesting side effects. Namely, although most directory user interfaces (found in browsers, mail programs) claim only to support person-related queries, in practise a user of the client could use the interface to send a query with role in the name entry.

Query with attribute Organisation

It is possible to send a query with attribute "organisation" but it would result in no hits because of that the organisation attribute is not included in the objectclass "inetOrgPerson". Roland Hedberg has proposed a change for the latest release of the objectclass definition document.

To provide the desired ability to narrow search focus to some range of organization names (attribute values), there are three possible approaches with differing merits/detractions:

Recommend the use of the "locality" attribute -- although a more standard definition would be required (locality is currently used for everything from organization to county to map coordinates).

Recommend or require that the attribute organisation should be inherited in objectclass "inetOrgPerson".

Build the LDAP DAG-SAP to submit 2 query to the WDSP. The second is the same as the first, with only cn filters if the entire query including "o" results in no hits (i.e., back off from the organization filtering if it doesn't seem to be supported).

Configuration

It is not possible to see what character set a LDAP clients want to use. The recommendation so far in he project has been to define a unique port for each character set. This requires extra end-user configuration of client software, and proper advertising of the port number-charset mapping provided in the service.

DN

When the user wants to look-up more information about a person found in a preliminary search, the LDAP client uses the entry's DN together with host and port to the DAG system. Not only does that mean that the client submits a non-compliant query to the DAG system, as DNs are not part of any of the defined queries for the service, it simply does not provide the desired effect of getting to the user's entry.

Response Codes

The LDAPv3 client that was used does not support more than 2 response codes -- "success" and "size limit exceeded". All the other response codes are translated to "size limit exceeded", although no results are returned. That is, if the error was in fact that the size limit was exceeded, the results up to the size limit are presented. If it was another response code mapped to that one, no results are presented.

Sending and loading CIP Index Objects

At least one server is quoting the CIP-object incorrectly for the Swedish characters A-Ring, A-Umlaut and O-Umlaut. Sending quoted printable CIP-objects with PINE mail software works.

Source - Labeled URI

The original plan for the use of the labeled-URI attribute was to use it to return a pointer to the WDSP that provided the user information. However, the standard use of the labeled-URI attribute, which may in fact be populated in the data returned by a WDSP, is to contain the URI for more private related homepages.

Full Copyright Statement

Acknowledgement

Funding for the RFC Editor function is currently provided by the Internet Society.