Service Location Protocol

Charles E. Perkins
Sun Microsystems, Inc.
15 Network Circle
Menlo Park, CA 94025
cperkins@eng.sun.com

Abstract

The Service Location Protocol has been designed within the Internet Engineering Task Force to meet the needs of all Internet nodes for automatic resolution of service needs. The protocol defines User Agents, Service Agents, and Directory Agents, and the request and response messages between them. Provisions are made for interactive selection of services. Scalable operation is provided by eliminating most multicast and broadcast messages, and by a scoping mechanism useful for dividing up the administration of enterprise resources into separate domains. The distributed and non-hierarchical nature of the service agent registration leads to further reductions in the need for administration, and to greater availability to local user agents. Security services are defined within the protocol to work in conjunction with other standard security services.

1 Introduction

The Service Location Protocol (SLP) [9, 3] has been designed to meet the need for automatic resolution of service needs in the enterprise and in the mobile environment. In the enterprise environment, this process for automatic resolution is driven by the wealth of heterogeneous resources as well as by the expense of administration. In the mobile environment, the number of resources available to the mobile user may be less, but the need for reducing administration is much more important. In many circumstances, the mobile user might not even be able to identify the right person to call. When SLP is in common use, telephone calls to the right system administrator, simply to perform routine operations will go out of style. Moreover, when a phone call is needed, the information available from the devices obeying SLP can include the right person to call whenever there is trouble.

SLP was inspired by the previous success of similar features in AppleTalk [2], which allowed interactive users to find network services automatically. Within the Internet, the lack of a similar degree of automatic configuration has always been viewed as one of the major points of difference (and inadequacy) of TCP/IP versus AppleTalk. In fact, SLP was designed within a working group of the Internet Engineering Task Force (IETF) that was started by former Apple employees who saw the same need for IP networks. Novell protocols also have offered similar autoconfiguration capabilities [8], and this has been a major reason for their continued popularity.

Within the Internet community, there has been one previous attempt at defining a similar protocol. At Carnegie Mellon University (CMU), the Resource Discovery Protocol (RDP) was defined for operation on local area networks. However, RDP made heavy use of multicast and broadcast, and it seemed unlikely to adapt well to larger environments. Moreover, as will be discussed in more detail, in order to make automatic configuration work a lot of attention has to be paid to defining conventional identifiers for the service types and the attributes which can be used to distinguish the several instances of service agents of each service type.

Since TCP/IP is used in a bewildering variety of enterprises and installations, SLP had to be designed with close attention to scalability. Experience with AppleTalk and Novell's protocols has shown that plans that work in small installations may not work as well in large enterprises. Conversely, designs that work in large enterprises but require deployment of expensive centralized facilities, are less likely to be appropriate for use in small office environments (for instance, a dentist's office). The variety of intended applications has also introduced a need for secure operation. We hope that SLP will find use for financial and commercial services as well as for more traditional network services such as printing and mountable filesystems.

The two main scoping mechanisms used with Service Location Protocol are the use of Directory Agents, and the ability to partition services into different subgroups identified by their "scope" name. Clients can multicast their needs and receive responses from remote services. In an enterprise, this requires wide area multicast and runs into scaling problems. A *Directory Agent* is used to centralize service information so clients can send directed queries without the need to multicast. This provides protocol scalability along the dimension of larger numbers of interconnected networks. To allow services to be deployed without overwhelming the abilities of local system administrators, SLP allows responsibilities and the usage of local services to be separated and divided up as appropriate. There is no need for any over-arching network or administrative hierarchy to keep track of every service that some employee might wish to introduce within the network.

The rest of this paper is organized as follows. Section 2 gives an overview of the protocol. Section 3 gives more details about specific protocol messages. It also describes the use of scopes within SLP, and how a network administrator might transition from one model of operation to another. Section 4 gives a short description of the use of new DHCP (Dynamic Host Configuration Protocol) options. Finally, section 5 concludes the paper and gives a few ideas about future directions.

2 Protocol Overview

The Service Location Protocol defines User Agents, Service Agents, and Directory Agents, as well as the basic protocol messages used by them. A User Agent (UA) operates on behalf of an application (which is itself running on behalf of some user, presumably) to acquire the information needed by that application for connection to a network service. Each network service is itself represented by a Service Agent (SA). A UA discovers an SA for a service of a given *service type* by transmitting a *Service Request*. In response, the UA expects to get a *Service Reply*.

When there are multiple networks, a UA will often wish to make a connection to a service on another network. To avoid excessive multicast, and to provide a common handler for Service Requests from UAs, SLP defines a Directory Agent (DA) that responds to such requests. The DA can receive the requests by unicast instead of multicast, and SLP defines several methods by which UAs can discover a DA appropriate for their needs, as detailed in sections 3.2.1 and 4 of this paper.

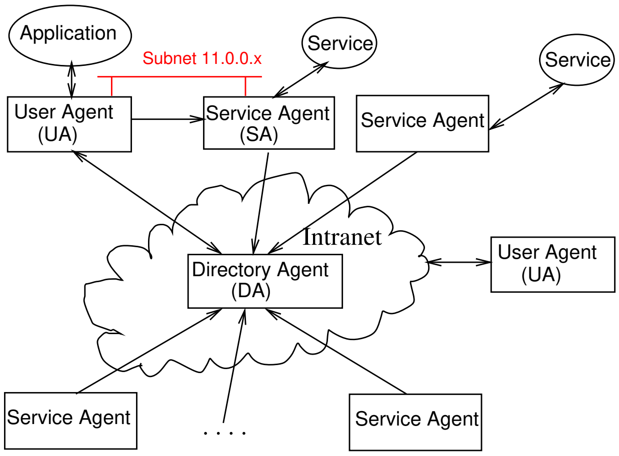

Figure 1: Overview of the base Service Location Protocol

Schematically, the operations just described are shown in figure 1. In the figure, an application and a service on net 11.0.0\24 can communicate without the need for any DA services from the rest of the Intranet, if they are so configured. In fact, SLP allows UAs and SAs on nearby networks to communicate without need for DAs, if they are configured to multicast Service Requests with TTL (Time to Live) greater than one. This may be appropriate in many installations that do not expect such multicast traffic to exceed a low percentage of their overall network traffic load.

The figure also shows that user agents can contact the DA for access to services anywhere else within the Intranet. The SAs on other networks *register* their services with the DA, and the UAs pick the services they need by using queries in a simple boolean query language. For additional scalability, SLP has been defined to work with many Directory Agents, and protocol operations are available so that SAs can select one or more particular DAs to register their services.

The protocol operations mentioned so far (request/response, and service registration) are sufficient for basic, noninteractive use of SLP by UAs, SAs, and DAs. There is also a specialized variation of the Service Reply known as a DA Advertisement. This advertisement can be multicast periodically by a DA¡ or it can be sent in response to a Service Request containing a query for *service type* `directory-agent`.

2.1 Interactive Use

For interactive use, there are protocol messages to enable a user to browse all the service types and service agents of each service type that happen to available to the user. These browsing messages provide the same information whether or not they are multicast to nearby networks, or on the other hand if they are directed at a suitable DA. It is also possible for a user to browse all the *service attributes* that are associated with available SAs. By looking at the service attributes (and, in the usual case, by looking at the descriptive text which further describes the purpose and range of values possible for the attributes) a user can interactively associate a particular SA to serve the transient needs of an application, or alternatively to configure their system to use that service by default. This mode of operation is likely to be of particular interest to mobile users as they enter a new domain of attachment to the Internet. As time goes on, and SLP is more widely deployed within most enterprises and across the global Internet, users may shed their dependence upon interactive operation because they can always be confident that their service needs will be met no matter what their current point of attachment to the Internet might be.

2.2 Zero Administration

From the point of view of the mobile and corporate user, SLP gives a way to achieve effortless and even transparent utilization for the network services needed by applications. This drive towards *zero administration* characterizes quite a lot of recent protocol design within the Internet. However, it is equally important to consider the administrative needs of the service provider, and of the network administrator. Here, too, SLP offers major benefits. One benefit is that SLP frees the corporate network management from the need to micromanage each service instance. For instance, a recent article in a trade magazine tells of network administrators with responsibility for more than 100,000 printers. Even if it is possible to discharge such a responsibility in an effective and diligent manner, it surely must not be pleasant. SLP offers the capability to reduce the workload for such administrators, by making the creation of service agents automatic, and by distributing the ongoing care of the network services to only those departments or individuals who are directly interested in making the services available in the first place.

2.3 Hierarchy – Not...

One element of this simplified administration is to remove any requirement for hierarchical organization of network resources. This is natural, because there is not necessarily any close mapping between the network resources available from one's laptop or desktop, and the address hierarchy, domain name hierarchy, or the organizational hierarchy of the administration offering the services. All of these hierarchies are, in fact, quite distinct, and no universal rules can be applied between any one of them and a useful model of service provision. SLP operates without any need for hierarchical administration. The other facet of SLP protocol operation that simplifies administration is the dynamic nature of the information supplied to Service Requests from UAs. A service agent will only answer a Service Request if the agent is in operation at the time the request is transmitted. DAs can have stale cache entries, but the lifetime of a registration from a SA is typically short enough that problems do not last too long. Moreover, since UAs receive replies that contain information about multiple SAs, the UA can try another one (perhaps even automatically) if the first one came from a stale registration.

2.4 Scopes

Services may be grouped together into subsets known as *scopes*. A scope is nothing more than a way to group together a collection of SAs; the semantics associated with the grouping is completely at the discretion of the network administrator defining the scopes. Typically, a scope might associate all the services paid for by a particular department of the campus or of an enterprise. User agents are then typically assigned to use services from a particular scope. This scope assignment can be done by way of DHCP (see section 4), or by static configuration of the UAs. It is expected that the former method will typically be preferred.

Scopes may be administered hierarchically, and this may often make sense in many enterprises. This hierarchical arrangement is up to the local administration, and would be enforced by administrative tools using techniques out of scope of the basic SLP protocol specification. During initial deployments, there may not be a need for any scope definition at all. When no scopes have been defined, SLP agents use a catch-all scope called the *default scope*, and include the string "`default`" in protocol messages requiring scope parameters. Then, as the deployment of SAs and UAs grows along with the enterprise, some of the agents will be assigned to particular scopes, while others continue to use the default scope. If there are multiple DAs, some DAs will respond to request for services in the default scope, and some will not. A DA or SA has to explicitly exclude the default scope from its configuration in order to respond to requests or registrations from the default scope, and this explicit exclusion can be done in a phased manner. In other words, the scope reconfiguration can be performed at each agent individually, without requiring any flag days. In this way, scopes can be introduced gradually without sudden dislocations of large populations of user agents.

2.5 Service Templates

In order for a UA to create a Service Request that enumerates appropriate *service attributes* used to select the right service, there has to be a commonly understood way of specifying those attributes. A service might supply values for the service attributes that match the values requested by a UA, but the match can only be detected if the UA and SA agree on the meanings of the *attribute tags* (i.e, the names by which the attributes are universally known).

The recipes that enumerate and describe these attribute tags are called *service templates*. Service templates include the following:

- the *service type*, which characterizes the general kind of service (e.g., "printer").
- the version number for the template
- the *language* in which the attribute names and values are given
- a description of the service type, and
- a list of attributes

For each attribute, the template specifies whether the attribute takes *integer*, *string*, *boolean*, or *opaque* (binary) values. Strings are made up of (possibly multi-byte) UTF-8 characters, and thus can include all printable ASCII characters. There are some reserved characters that are used to formulate the boolean syntax of the Service Request attribute query. Each attribute can be further specified to be *optional*, *multi-valued*, *literal*, or *expected*. Optional attributes can have default values. Service agents are allowed to give more than one attribute value for multi-valued attributes. Literal attributes cannot be translated into other languages, which makes them unusual because Service Requests can generally be generated with attribute values translated into whatever language the local administration deems appropriate. Finally, an attribute marked as an expected attribute *should* be a part of any Service Request, or otherwise the UA may receive a Service Reply containing information which it really cannot use. A typical example of this latter case is when a UA needs to search for a server that has user-specific files, like an IMAP server for e-mail.

Service templates are standardized by following a procedure defined by the `svrloc` working group within the IETF. The templates are made universally available by the actions of the Internet Assigned Number Authority (IANA). However, it is possible for specific enterprises to define alternative service types for use in a restrictive domain. This is done by designating that the service type defined by the template is under the control of an alternative *naming authority*.

2.6 Security

For functions such as access control, privacy, and authentication for specific UAs or SAs, SLP relies on the use of IPsec protocols as recently promulgated within the IETF [4, 5, 6]. However, IPsec works to provide security on the basis of the IP addresses assigned to nodes, and part of the point of SLP is to take away the need for the UA to obtain services based on specific IP addresses. Instead, using SLP, a UA can obtain service just by knowing the desired service type and the attributes of the desired service; any such service will do no matter what its IP address happens to be. Nevertheless, in many situations the UA may need to be sure that a SA is *authorized* to provide the requested service. In SLP, a service is known to be authorized if it is a member of a *protected scope*; a service in a protected scope supplies a unforgeable checksum along with its service information during registration. This checksum is authenticated by UAs before making use of the service information.

3 Protocol Details

SLP messages use UDP or TCP as the transport protocol, and are received on port 427 in either case. The source port is arbitrary. SLP entities (UAs, SAs, and DAs) have to be able to handle both transport protocols. SAs and DAs also have to join a multicast group and listen for protocol messages at a well-known multicast address.

Messages in SLP can be divided into two groups: the messages which are mandatory for UAs, SAs, or DAs, and the optional messages. Typically, the mandatory messages are those which are useful in non-interactive operations; they are mandatory exactly because without them, no SLP entity could really expect to fulfill its function in an interoperable way. The optional messages are typically useful for interactive browsing and selection of local network services. The mandatory messages also useful for interactive use, but mainly because they form the basis for the whole protocol, not because of the particular nature of the interactive process.

Every SLP message has a message header with the common fields illustrated in figure 2. The length

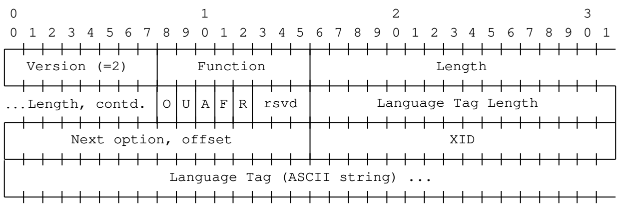

Figure 2: SLP common message header

is 24 bits to allow for *very* long service reply messages, which might have information about numerous service instances, each supplied with authentication data that can be quite lengthy The `XID` field identifies a request or registration message, so that it can be matched up with the corresponding reply or acknowledgement. The flags serve specialized purposes, some of which will be described later. However, the 'O', 'U', and 'A' flags are general purpose. The first indicates message overflow in a UDP response message, and suggests that the corresponding request be reissued using TCP. The last two flags indicate the presence of authentication data for a service: URL or attribute data, respectively. The use of service: URLs will be described later in section 3.1.2.

3.1 Mandatory Service Location Protocol Messages

The mandatory messages are:

- Service Request
- Service Reply
- Service Registration
- Service Acknowledgement
- Service Deregistration

- Directory Agent Advertisement

Given the protocol overview in section 2, the function of these messages is largely self-explanatory. For instance, UAs are required to be able to send Service Requests and receive Service Reply messages. DAs are required to be able to receive Service Requests, Service Registrations, and Service Deregistrations. They are further required to be able to formulate and send Service Reply messages as well as Service Acknowledgements.

3.1.1 Service Request Message

A Service Request message specifies the service type and and the service attributes needed by a UA. The service type is a well-known name for a kind of service, for instance "printer". The attributes for the printer service type are items like the fonts supported by the printer, whether the printer supports color printing, or PostScript, or has three-hole paper loaded. These attributes are combined into a query called a *Service Request predicate*, by using boolean connectives and parentheses with a prefix notation. Certain comparators are also supported for integer attributes, for example '<', '>', and so on. So, for instance, one might see a query of the form //

$$(\&(attr1 = some\_string)(attr2 >= 33))$$

The Service Request also contains the name of the scope from which the service instance should be chosen. This scope name has to be specified, but may take the special value `default`. Lastly, the 'R' (*request multicast*) flag may be set in the common protocol header (see section 3) when the Service Request is multicast instead of unicast. This affects processing in cases when the same computer hosts both a DA and a SA.

In cases when the service type is locally defined, the service type string in the Service Request can be modified by appending a *naming authority* at the end of the type string, delimited by a '.'. Thus, to obtain a printer with a service type defined locally by the network administration of ABC corporation, the service type string `printer.ABC` might be used in the Service Request, to indicate the use of the ABC naming authority. Such service types are not necessarily kept track of by IANA, and therefore would not be assumed to be available outside the network administration of ABC corporation.

3.1.2 Service Reply Message

A Service Reply transmits information about service instances to UAs in response to a Service Request. The information is sent in URL entries, and each URL follows the rules for schemes specified in [1]. A special class of *service schemes* has been initiated for use with SLP, each starting with the string `service:`. URLs following the recipe for service schemes in the Internet have the following format:

$$service :< service\_type >: // < addr > / < data >$$

A URL entry augments each service: URL with a lifetime and, in some cases, with authentication information. The $< addr >$ field of the URL is usually a *fully-qualified domain name* (FQDN) [7] for the IP addressable computer hosting the service instance along with possibly a port number (and, technically, a user *login name* can be specified)

Recently, schemes for address families other than IP have been defined. Such schemes are indicated by inserting the alternative address family name (i.e., `AppleTalk` or IPX) between the first two '/' characters after the service type name. URLs defined by schemes of alternative naming authorities also append the naming authority to the service: URL, delimited by a '.' character.

3.2 Service Registration Messages

There are three messages related to registration of service instances, namely the registration and deregistration messages, and the acknowledgement message. The registration message includes all the field necessary to allow a DA to determine whether a particular service instance matches the needs expressed by a UA in a Service Request. This includes the following:

- the URL entry for the service, with lifetime and possibly authentication

- The service type name
- the scope to which the service instance belongs
- the list of attribute values describing the service instance, and
- for services in protected scopes, authentication for the attribute values.

Incremental registrations are allowed; that means that service registrations for already registered services are allowed to specify updated values for particular attributes without having to send in a complete list of all attributes and scopes, etc. In the case of incremental registration, the common protocol header (see section 3) does not have the 'F' (*fresh*) flag set. When the 'F' flag is set, the new registration completely replaces the previous registration, even when all the attributes of the old registration are not re-specified in the new one. Because of difficulties with authenticating incremental registrations, they are not allowed for services in protected scopes.

Services can be registered in multiple languages. A Service Request asks for services that match attributes, and the attributes can be specified in a number of languages (default, English). If the request is sent to a DA, the DA tries to match the attribute values using a registration in the appropriate language. If no registration for a service of the requested type has been received in the requested language the DA is allowed to attempt to translate the attribute values from the registered language to the requested language before trying to make the match. When the DA has access to service templates standardized for both languages, the translation can be perfect; otherwise, the DA may attempt a "best effort" translation.

A Service Acknowledgement is sent to a SA in response to every Service Registration (or a Service Deregistration). If the SA does not receive the acknowledgement, it is required by protocol to try the registration again until either the acknowledgement is received, or the SA can be sure that the DA is no longer reachable.

3.2.1 DA Advertisement

The third type of required message is the special SLP message by which DAs can make themselves known to UAs and SAs. This message is almost like a Service Reply, except that the URL used for locating the DA is accompanied by a timestamp that changes only when the DA reboots. SAs that see a new boot timestamp in the DA Advertisement are required to register their service instances again, since the DA may no longer have any record of the previous registration. The URL in the advertisement has the form:

```
service:directory-agent://<addr>
```

DAs typically transmit such advertisements at some regular interval, so that new SAs in the general area may be expected to discover the DA, and so that existing SAs may be able to tell when the DA has rebooted.

A DA can also unicast an advertisement in response to a multicast Service Request for service type `directory-agent`. The advertisement can only be sent when the scope in the request matches a scope configured for use with the DA. For further DA discovery modes, see section 4.

3.3 Optional Messages

The optional SLP messages are:

- Service Deregistration
- Service Type Request
- Service Type Reply
- Attribute Request
- Attribute Reply
- SA Advertisement

A Service Deregistration message is very much like Service Registration message, except that the attributes deregistered to not have to have any attribute values. Like a registration, a deregistration can be incremental, which means that it may deregister only certain attributes of the service instance without completely deregistering the service.

The Request/Reply messages for service attributes and service types are largely designed to facilitate the construction of advanced user interfaces and browser software. The major motivation for using service: URLs as the basic locator syntax for services in SLP was to simplify the use of Web browsers for locating network services. Besides that, however, it was observed that designing Internet locators is a task that is tricky and best done using the experiences gained by others. There is no reason to repeat the mistakes of the past, especially when it comes to writing specifications for reserved characters, internationalization, alternate address families, and lexical methods for identifying hosts vs. ports vs. user names vs. pathnames and so on.

3.3.1 Attribute Request and Reply

A UA may send an Attribute Request to a DA, and receive all the attribute values that are available for a particular service instance, or for a particular service type. To specify a particular service instance, the UA includes the service: URL for that service. To get attribute values for all instances of the service type, the UA deletes the $< addr >$ and $< data >$ fields that would normally be part of the service: URL for a particular instance. Attribute values are only provided for services that belong to the scope requested by the UA.

An Attribute Request may also be multicast. In this case, all SAs that receive the request will send back their attribute information to the requesting UA, and no information will be sent by any DAs that happen to receive the request. In this situation, there is a danger that some of the responses from SAs might be lost if there are a lot of them being transmitted all at once. To reduce this danger, the UA should continue to multicast the request with a completed PRList (*Previous Responders* List) field, until there are no more replies coming. The PRList field contains the URLs for SAs that have already answered; these SAs are required to ignore subsequent requests with the same transaction ID (XID).

For attributes registered by a service instance in a *protected scope*, the Attribute Reply will contain authentication information to be verified by the UA, whether or not the request was multicast.

3.3.2 Service Type Request and Reply

The Service Type Request message allows a UA to find all the service types available to it, possibly restricted to those defined by a specified naming authority. If no naming authority is specified, IANA is assumed. The UA also has to specify the scope it is interested in, and only service types corresponding to service instances registered in that scope will be returned in the reply message.

Service Type Request messages, like Attribute Request messages, can be multicast. For similar reasons, they can also contain PRLists.

4 DA and Scope Configuration with DHCP

Recent work has resulted in new Dynamic Host Configuration Protocol (DHCP) options 78 and 79 for use with SLP. Option 78 is used to configure a UA or a SA with the address of a DA and scope. Option 79 is used just for configuring SLP agents (UAs, SAs, or DAs) with scope information. Only UAs and SAs can use option 78.

Getting a DA address from option 78 is straightforward, but complicated somewhat because DHCP deals with hosts on a per-address basis. SLP, on the other hand, allows multiple UAs, SAs, and even a DA to co-exist on the same IP host. To cope with this, the syntax for option 78 allows DHCP to specify different scopes for each of the several possible service types on a host. The restriction is made that every service instance for any particular service type must be in the same scope.

When a UA or SA requests an IP address from DHCP, it should at the same time request the necessary scope information and the address of a DA, if available. Scopes are designed for maximal convenience to the network administrator, and in a properly designed installation would be almost completely invisible to a human using the SLP-enabled machine. The main exception to this rule is that DAs periodically send out advertisements containing the scopes handled by the DA, but this information is typically not needed for any user interface handling or applications being served by the UA software. Use of DHCP to obtain scope information by the SLP agents (in either option 78 or 79) goes along perfectly with this philosophy. Furthermore, using DHCP in this way enhances scalability, since very little additional network load is required for the scope configuration.

Including the above described DHCP option 78, there are in total four ways for UAs and DAs to get DA addresses:

- Static configuration
- DHCP option 78
- Periodic DA advertisement
- Response to a Service Request with the service type `directory-agent`

Any SA or UA implementation should be able to operate with any or all of the four modes of discovering DAs, in the order just listed.

5 Summary and Future Work

SLP offers a flexible and powerful solution to the problem of automatic resolution of network service needs, a problem of major importance for mobile networking. This automatic facility will enable a new degree of freedom and productivity for mobile users as they connect to new points of attachment to the Internet. SLP offers a scalable protocol designed to serve the needs of enterprises large and small. SLP eliminates almost all need for user administration of the mobile computer for access to network services, while at the same time reducing the workload of the network administrator substantially. The scope mechanism, the distributed maintenance of service information in Directory Agents, and the lack of required hierarchical control of service instances all contribute to the ease of administration inherent in SLP.

Future work will involve the extension of SLP into the global Internet as well as gaining experience with the deployment of SLP in larger enterprises. As the population of DAs grows in an enterprise, it is likely that DAs should learn how to share information to increase the availability of service instances and robustness of the service registrations. Recent developments with availability of source code for browsers may also lead to a higher degree of integration between browser interfaces and SLP features. Lastly, there are likely to be multiple points of contact between SLP and other directory services such as LDAP and `whois`. Working out the most productive and beneficial cooperation between directory services is a project that will probably go on for many years, for better or for worse.

References

[1] T. Berners-Lee, L. Masinter, and M. McCahill. Uniform Resource Locators (URL). RFC 1738, Dec. 1994.

[2] S. Gursharan, R. Andrews, and A. Oppenheimer. *Inside AppleTalk*. Addison-Wesley, 1990.

[3] E. Guttman, C. Perkins, J. Veizades, and M. Day. Service Location Protocol version 2. draft-ietf-svrloc-protocol-v2-04.txT, Mar. 1998. (work in progress).

[4] S. Kent and R. Atkinson. IP Authentication Header. draft-ietf-ipsec-auth-header-03.txt, Nov. 1997. (work in progress).

[5] S. Kent and R. Atkinson. IP Encapsulating Security Payload (ESP). draft-ietf-ipsec-esp-v2-02.txt, Nov. 1997. (work in progress).

[6] D. Maughan, M. Schertler, M. Schneider, and J. Turner. Internet Security Association and Key Management Protocol (ISAKMP). draft-ietf-ipsec-isakmp-08.txt, July 1997. (work in progress).

[7] P. Mockapetris. Domain Names - Concepts and Facilities. IETF STD 13, Nov. 1987.

[8] I. Novell. Advanced netware v2.1 internetwork packet exchange protocol (ipx) with asynchronous event scheduler (aes), Oct. 1986.

[9] J. Veizades, E. Guttman, C. Perkins, and S. Kaplan. Service Location Protocol. RFC 2165, July 1997.