A. Introduction

The Service Tag Discovery protocol provides a built-in way for clients to discover where the Service Tag listener is running. By default, the in.stdiscover daemon will listen for discovery probes (using a minimal built-in protocol) on UDP port 4950.

This built-in discovery protocol is intended to be the default, but not exclusive mechanisms for exposing Service Tag information to a network. In many cases, it will be desirable to use standardized discovery protocols (such as SLP or DNS-SD) in addition to or instead of the default. Service Tags will support such choices wherever possible. Those alternatives are not being adopted as the default simply because they are heavier-weight (being generalized by design) and thus would be difficult to universally support on the variety of target platforms for Service Tags (everything from full-OS instances of Solaris, Linux, HP-UX, and Windows to minimal OS instances for embedded service processors).

Note that in most cases, the discovery software is expected to be running with the "noaccess" uid and gid.

The following sections describe the default discovery protocol.

B. Discovery Protocol

The built-in discovery protocol for Service Tag agents uses UDP packets (on port 4950, by default). Requests (usually from a Service Tag "Collector" component) may be unicast or broadcast. Multicast may also be supported in the future, but has not yet been fully investigated. The format for a discovery request (or "probe") message and the expected response to such a message are described in this section.

B1. [PROBE] ${unique_cookie}

Purpose

Used to discover the existence of Service Tag agent instances. The ${unique_cookie} value should be some random string which can be used by recipients to detect rebroadcasts.

Response Description

Returns a message indicating that a Service Tag agent is available on the responding machine. The response format is "${cookie} ${agent_urn} ${port} ${base_uri}" (which indicates that a response to the probe message which contained the specified cookie, saying that the agent with the specified URN is available on that port and base URI). The IP address for which the port applies must be obtained from the response message's own IP header. (The intent of the latter is to partially protect against the possibility of malicious responses directing traffic to a 3rd party, as well as hopefully making this protocol more usable in networks employing NAT--where the IP header will be rewritten but the proprietary payload almost certainly will not.)

Note also that responders are expected to sleep for a random period (currently set to between 0 and 1/10th of a second) prior to returning any response. The intent of this mechanism is to protect against unintended flooding of the requester (such as in cases where a probe was sent to a large broadcast domain containing many agents).

Example Response

```
cookie-val-123-456 urn:stagent:123-456 4950 /stv1
```