Internet Draft
Document: draft-beck-rescap-req-02.txt
June 15, 1999
Expires December 15, 1999

John Beck
Sun Microsystems

ResCap Requirements

Status of this memo

Abstract

A variety of resource identifiers have been widely deployed on the Internet as a means of identifying various resources, services, and destinations. However, a means of attaching a set of attributes or characteristics to a given resource identifier and subsequently assessing those attributes or characteristics has not been specified and deployed.

A particularly important resolution service of this general type is one which, when given a mail address identifying a particular mail recipient, will return a series of attributes describing the capabilities of that recipient. This differs from a directory service in that no searching or other advanced query operations are involved. Likewise; this is not a discovery protocol.

Two tasks are envisioned. The first task will be to define a general resolution protocol that will translate resource identifiers to a list of attributes. The second task will be to define an administrative model and update protocol that can be used to set up and maintain the information the resolution protocol accesses.

This document defines the requirements for these two protocols.

0. Discussion

This draft is being discussed on the ResCap mailing list at <rescap@cs.utk.edu>. Subscription requests can be sent to <rescap-request@cs.utk.edu> (send an email message with the word "subscribe" in the body). More information on the mailing list along with an archive of back messages is available at <ftp://cs.utk.edu/pub/rescap>.

1. Resolution protocol requirements

Throughout the rest of this section, "the protocol" refers to the resolution protocol.

1.1 Scalability

The protocol must be highly scalable both for number of entries in the database and number of entries per second resolved.

Example: Mail services with tens of millions of users could easily expect tens of millions of requests per day for client attribute information.

1.2 Reply data

The protocol should be capable of returning resource capabilities that are arbitrarily long text or binary values. (E.g., Conneg [CONNEG] uses arbitrary length text values; public key certificates are arbitrary length binary values; etc.) Such data might overflow a UDP datagram, so the protocol must allow for this; however, the default should be to use UDP. [UDP]

Example: Lists of media features or S/MIME certificates can easily be longer than a single UDP datagram.

1.3 Granularity

A mechanism needs to exist whereby a subset of capabilities for a resource can be fetched. I.e., the protocol request syntax should be able ask for one or more features instead of all of them at once. However, the client also needs to be able to ask for all capabilities known to the server without naming all of them.

Example: A client might only want to know the S/MIME capabilities of a recipient, but not care about its media features.

1.4 Expiration

Some means to indicate the expected lifetime of a capability is required, so that a client application can judge whether, or when, the information should be considered stale.

The protocol should also support a mechanism for indicating the "last changed date" of a given attribute.

Example: The server may believe that the recipient is only temporarily unable to receive large mail messages.

1.5 Referral

Some sort of request referral mechanism is needed. In other words, the protocol must support a mechanism whereby a response can indicate "I don't know, but go ask the ResCap server at address X." or "use the following URL to retrieve the ResCap response you requested." That is, the response might be a simple DNS name, or it might be a full ResCap URL.

Example: A server might delegate all requests for S/MIME certificate information to a different server that keeps track of that type of information.

1.6 Security

The protocol must be able to handle authenticated queries. The protocol must also support transmission of signed and/or encrypted responses. The protocol should allow for a server to "pre-sign" responses, meaning that the server could sign part of a response off-line so it could present this over and over. Controls on which attributes will be announced should exist.

Example: A server might give less information to a client that is unauthenticated than to one that is authenticated. Some information from the server may be important enough for the server to want to prevent tampering, or even to prevent snoopers from reading.

1.7 Server location

SRV and/or NAPTR resource records may be used to determine a protocol server. [SRV, NAPTR]

Example: The ResCap server that is running on the host "example.com" might not be the ResCap server for all resources that have the host "example.com", such as if the administrator at "example.com" had outsourced ResCap services for some resource types to another company.

1.8 Preference

For a set of capabilities, there should be a means to indicate a preferred value or a ranking of preference.

Example: A recipient might strongly prefer image/tiff files over image/jpeg because s/he can display image/tiff on his/her system without launching an external application.

1.9 Simplicity

The protocol should be sufficiently simple that it allows implementation of client and/or server functionality in very small, low cost devices (e.g. telephones, modems, printers, smart-cards, etc.).

2. Administrative update protocol requirements

Throughout the rest of this section, "the protocol" refers to the administrative update protocol.

2.1 Access control

Authentication of anyone updating the database is required.

Example: Individual mail users should be able to update some or all of the information about them in the database, but such updates must be done with authentication to prevent others from maliciously entering false information.

2.2 Inheritance

The protocol must support inheritance. Specifically, mechanisms must be provided by which administrators can set default values for members of their administrative domains.

Example: The media features for all addresses at a particular mail server might be the same because the mail server processes all messages at all addresses.

3. Security Considerations

Security issues are discussed in sections 1.6 and 2.1 of this memo.

4. Acknowledgements

The author would like to thank Paul Hoffman and Graham Klyne for their assistance.

5. References

[CONNEG] "A Syntax for Describing Media Feature Sets", RFC 2553.

[CONNEG-MEDIA] "MIME content types in media feature expressions", draft-ietf-conneg-feature-type.

[NAPTR] "Resolution of Uniform Resource Identifiers using the Domain Name System", RFC 2168.

[SRV] "A DNS RR for specifying the location of services (DNS SRV)", RFC 2052.

[UDP] "User Datagram Protocol", RFC 768.

6. Author's Address

John Beck
Sun Microsystems
901 San Antonio Road
M/S U-MPk-17-202
Palo Alto, CA 94303-4900
(650) 786-8078
jbeck+rescap@eng.sun.com