Network Working Group
INTERNET-DRAFT
Intended Category: Informational
Expires: April 1999
Filename: draft-lachman-ldap-mail-routing-03.txt

H. Lachman
Netscape Communications Corp.

October 1998

LDAP Schema Definitions for Intranet Mail Routing - The mailRecipient Object Class

Status of this Memo

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

This document is an Internet-Draft. Internet-Drafts are working documents of the Internet Engineering Task Force (IETF), its areas, and its working groups. Note that other groups may also distribute working documents as Internet-Drafts.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as ``work in progress.''

To learn the current status of any Internet-Draft, please check the ``1id-abstracts.txt'' listing contained in the Internet-Drafts Shadow Directories on ftp.is.co.za (Africa), ftp.nordu.net (Europe), munnari.oz.au (Pacific Rim), ftp.ietf.org (US East Coast), or ftp.isi.edu (US West Coast).

Copyright Notice

Abstract

Directory services based on the Lightweight Directory Access Protocol (LDAP) [1] and X.500 [2] provide a general-purpose means to store information about users and other network entities. One of the many possible uses of a directory service is to store information about users' email accounts, such as their email addresses, and how messages addressed to them should be routed. In the interest of interoperability, it is desirable to have a common schema for such email-related information.

This document defines an object class called 'mailRecipient' to

support SMTP [3] message transfer agents (MTAs) in routing RFC 822-based email messages [4] within an organization. The intent is to suggest a model for MTA interoperability via the directory, to provide information about a solution that has been implemented and deployed, and to stimulate discussion about whether and how to standardize the functionality in question.

1. Background and Motivation

LDAP-based directory services are currently being used in many organizations as a repository of information about users and other network entities (such as groups of users, network resources, etc.). Some information is stored in the directory for the consumption of persons browsing for information (e.g., telephone numbers, postal addresses, secretary's name). Other information (e.g., login name, password, disk quota) is stored for use by one or more network applications or services. This latter use of the directory suggests the opportunity to centralize the storage and management of user account information related to different services. In general, it is advantageous for different network applications and services to refer to the directory for user account information, rather than each service keeping its own collection of user account records, which requires the network administrator to separately create or destroy user entities, passwords, etc., in many different systems each time a user joins or leaves the organization. The goals of centralized user management and sharing of information with other service types drove our decision in the design of Netscape Messaging Server (an SMTP-based mail server product) to use LDAP-based directory services as a common repository for user account information.

Thus, in our implementation, all account information for a given mail server user is stored in the directory entry that represents that user. This includes the user's delivery options, access restrictions, mailbox quota, and vacation status, among other things. Now, if a given mail server can refer to the directory for its own users' account information, it follows that that same information can be made visible to other LDAP-aware mail servers in the same organization, and therefore that information can aid those other mail servers in correctly routing messages to users of the mail server in question. This assumes that there is an agreed-upon set of per-user attributes to support message routing among the mail servers in the organization. If this assumption is met in our implementation, we can obviate other means currently employed to specify per-user

message routing (such as the sendmail "aliases" database). The benefit of this is to further consolidate per-user system information.

If different vendors provide LDAP-aware mail server products, each

having its own schema for message routing, then the above benefits can be achieved for single-vendor customers, but customers who have multiple vendors' mail server products would not be well served. They will likely expect interoperability, which will require a common schema to be supported by the various vendors' products. Thus, it is worthwhile to consider how to develop a common schema.

This document defines a schema designed to provide a means by which a directory entry that represents a mail recipient can provide information enabling MTAs to route messages to the recipient's "home" MTA. This document considers only intra-enterprise SMTP message routing using LDAP-based directory services. Solutions and issues involving inter-enterprise routing, non-SMTP message handling, non-LDAP directory services, and other messaging management topics not related to message routing, are outside the scope of this document (except that the concepts presented may also be applicable in the case of any X.500-based directory service).

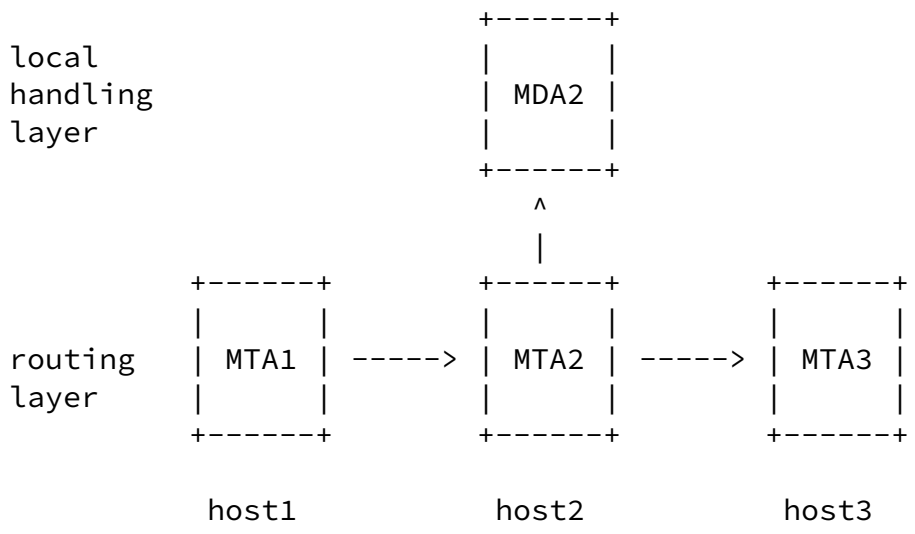
2. Overview of the Approach Implemented

In our design of Netscape Messaging Server, we identified all pieces of per-user account information, and assigned attributes such that the information for a given user can be held in the user's "LDAP entry" (the directory entry representing the user in an LDAP-based directory service). We segregated the attributes into two subsets: those that are of interest only to the "target MTA" (i.e., the MTA that considers the recipient to be local), and those that are of interest to "intermediary MTAs" (i.e., MTAs that are not the target MTA). Each subset of attributes is aggregated into an object class, the former being 'nsMessagingServerUser' (see Appendix), and the latter, 'mailRecipient'. It is the latter object class that is the focus of this document.

The 'mailRecipient' object class provides attributes that may be used to specify addressing and routing information pertaining to a given recipient. This information may be used by an intermediary MTA to

route a message to the recipient's designated target MTA, i.e., to the MTA that "takes responsibility" for messages to the recipient in question. The target MTA then accepts the message and, regarding the recipient as local, handles the message as specified by attributes intended for use by the target MTA (such as those associated with the 'nsMessagingServerUser' object class).

Consider a network with three hosts that run MTAs:



The above illustrates a two-layer mail routing and delivery model. The attributes provided by the 'mailRecipient' object class are used by the lower layer (the routing layer) to support the routing of a message to the correct target MTA. Other attributes may be used by the upper layer, which roughly equates to what is commonly called an MDA (message delivery agent), although the local handling may or may not involve delivery of the message to a mailbox (e.g., the message may be resent if the recipient is a mail group or a forwarded account). (In this discussion, "target MTA" means "target Messaging Server" which includes both MTA and MDA functionalities; while the implementation is not necessarily layered internally as implied above, the product nonetheless exhibits the functionality described.)

In our implementation, an LDAP entry that represents a mail recipient will have two mail-related object classes, 'mailRecipient', plus an additional one that may be used by the local handling layer to determine the recipient type and how messages for the recipient are to be handled on the target MTA. A mail user account will have 'mailRecipient' plus 'nsMessagingServerUser'. A mail group will have 'mailRecipient' plus 'mailGroup' [5]. An MTA need only look at attributes associated with 'mailRecipient' to determine whether a recipient is local, and if not, how to route the message. The additional object class and attributes are of interest only if the recipient is local.

(Note: While the Messaging Server fully implements this approach, earlier versions of its account creation tool do not place all of the above-mentioned object classes in the entries it creates. The Messaging Server is compatible with both the old and the new object class interpretations.)

A Netscape Messaging Server can route messages to recipients on other vendors' MTAs if the users' LDAP entries have the 'mailRecipient' object class and associated attributes. (Other vendors' MTA implementations may or may not follow the above-described model of indicating recipient type and MDA-level account configuration in LDAP, since only 'mailRecipient' and associated attributes are required for MTA-level recognition.)

Likewise, other vendors' MTAs can route messages to recipients on a Netscape Messaging Server if they recognize and interpret the 'mailRecipient' object class and associated attributes as defined in Sec. 3.

The intent of this model is to provide a framework within which any vendor can define new types of mail recipients, without requiring other vendors' implementations to have knowledge of the new recipient types; they need only have a consistent interpretation and application of the 'mailRecipient' object class and associated attributes.

In short, the main advantage of the 'mailRecipient' object class is to define a single object class that can serve to identify an LDAP

entry as an entity to which email can be addressed, and to aggregate the attributes that can provide multivendor MTA interoperability via the directory.

3. Object Class and Attribute Definitions

The 'mailRecipient' object class and associated attributes are defined (using syntaxes given in [6]) as follows.

3.1 The mailRecipient Object Class

```
( 2.16.840.1.113730.3.2.3
    NAME 'mailRecipient'
    SUP top
    AUXILIARY
    MAY ( cn $ mail $ mailAlternateAddress $
        mailHost $ mailRoutingAddress
    )
)
```

The 'mailRecipient' object class signifies that the entry represents an entity within the organization that can receive SMTP mail, such as a mail user account or a mail group account (mailing list).

The 'cn' attribute (common name) is provided as a means for administrators to identify the entry [7].

3.2 Address Attributes

```
( 0.9.2342.19200300.100.1.3
    NAME 'mail'
    DESC 'RFC 822 email address of this recipient'
    EQUALITY caseIgnoreIA5Match
    SYNTAX '1.3.6.1.4.1.1466.115.121.1.26{256}'
    SINGLE-VALUE
)
```

The attribute name 'mail' is a synonym for 'rfc822Mailbox', as defined earlier in [8]. This attribute specifies the recipient's "primary" or "advertised" email address, i.e., that which might appear on a business card; for example, "user@example.com".

```
    ( 2.16.840.1.113730.3.1.13
        NAME 'mailAlternateAddress'
        DESC 'alternate RFC 822 email address of this recipient'
        EQUALITY caseIgnoreIA5Match
        SYNTAX '1.3.6.1.4.1.1466.115.121.1.26{256}'
    )
```

The 'mailAlternateAddress' attribute is used to specify alternate email addresses, if any, for the recipient; for example, "nickname@example.com".

When determining the disposition of a given message, an MTA may search the directory for an entry with object class 'mailRecipient' and a 'mail' or 'mailAlternateAddress' attribute matching the message's recipient address. If exactly one matching entry is found, the MTA may regard the message as being addressed to the entity that is represented by the directory entry.

In short, address attributes may be used by an LDAP entry to answer the question "what is/are this account's email address(es)?"

3.3 Routing Attributes

```
    ( 2.16.840.1.113730.3.1.18
        NAME 'mailHost'
        DESC 'fully qualified hostname of the SMTP MTA that
            handles messages for this recipient'
        EQUALITY caseIgnoreIA5Match
        SYNTAX '1.3.6.1.4.1.1466.115.121.1.26{256}'
        SINGLE-VALUE
    )
```

The 'mailHost' attribute indicates which MTA considers the

recipient's mail to be locally handlable. This information can be used for routing, in that an intermediary MTA may take it to be the destination for messages addressed to this recipient.

```
    ( 2.16.840.1.113730.3.1.47
        NAME 'mailRoutingAddress'
        DESC 'RFC 822 address to use when routing messages to
            the SMTP MTA of this recipient'
```

```
        EQUALITY caseIgnoreIA5Match
        SYNTAX '1.3.6.1.4.1.1466.115.121.1.26{256}'
        SINGLE-VALUE
    )
```

The 'mailRoutingAddress' attribute indicates a routing address for the recipient. An intermediary MTA may use this information to route the message to the MTA that handles mail for this recipient.

Only one of the above two attributes need be present in order to route messages on behalf of the recipient. The 'mailRoutingAddress' attribute is more explicit in terms of providing an address that can be used to rewrite the SMTP envelope recipient address. With 'mailHost', the envelope address either is not rewritten, or is rewritten according to implementation-specific rules and/or configuration.

In short, routing attributes may be used by an LDAP entry to answer the question "how should MTAs route mail to this account?" (analogous to using the sendmail "aliases" database for per-user routing within an organization). This is in contrast with "forwarding" (see Appendix); forwarding and delivery options may be used by an LDAP entry to answer the question "what happens to mail once it arrives at this account?", which may include forwarding to some other account within or outside the organization (analogous to using the sendmail ".forward" file).

4. MTA Implementation Details

This section provides details of the algorithms followed by the Netscape Messaging Server as they relate to the 'mailRecipient' object class and associated attributes. Our implementation includes features that go beyond what is minimally needed to support the schema defined in Section 3, and other MTA implementations need not match our implementation in every detail in order to be interoperable (especially since various features described here can be disabled); but, in general, the features described here are recommended.

4.1 Finding the LDAP Entry for a Given Email Address

When the MTA receives a message, it attempts to determine whether

there is an LDAP entry that represents the recipient. It takes the SMTP envelope recipient address, and performs a search in LDAP, spanning the directory subtree specified in the configuration, for an entry that has the object class 'mailRecipient', and has either a 'mail' or 'mailAlternateAddress' attribute matching the recipient address in question. If exactly one match is found, this is taken to be the LDAP entry that represents the recipient.

If there were zero matches, but the domain part of the address matches the local MTA's hostname, we perform a fallback search with the same address except that the domain part is truncated to not include the host part (e.g., the search for "user@nsmail1.example.com" is retried as "user@example.com"). This fallback search is optional, as per the server configuration.

If there were zero matches so far, but the domain part of the address is considered to be local (by configurable criteria), we perform a fallback search for an LDAP entry that has object class 'mailRecipient' and a 'uid' attribute (i.e., login name; synonym for 'userid' [8]) equal to the local part of the recipient address. This fallback search is optional, as per the server configuration.

If the MTA finds the LDAP entry representing the recipient, it proceeds with the logic discussed in Section 4.2. Otherwise, it will rely on other information resources to determine whether to reject the message or route it elsewhere.

Note that LDAP entries without the 'mailRecipient' object class are ignored (except as may be needed for backward compatibility). This is necessary because some sites have LDAP entries that do not represent mail recipients, but have a 'mail' attribute nonetheless. For example, a conference room might have an LDAP entry including an email address, telephone number, etc., that are the same as for the secretary who books reservations for the room. In this example, the conference room's email address is for contact information only, and is not intended to imply that it has an email account. Therefore, the MTA correctly ignores the conference room's LDAP entry, and avoids producing multiple matches on the search.

4.2 Deciding Whether a Message can be Handled Locally

If the MTA has found the LDAP entry representing the recipient, as per Section 4.1, it checks the LDAP entry's 'mailHost' value to see if it matches the MTA's local hostname. If so, it handles the message locally. (Note that since accounts hosted on a Netscape MTA are expected to have a 'mailHost' value, they typically do not have a 'mailRoutingAddress' value; other implementations could make

different design choices, and still be compatible.)

Otherwise, it routes the message as specified by the 'mailHost' value and/or the 'mailRoutingAddress' value. See Section 4.3 for further details.

If the recipient's LDAP entry contains no routing information (i.e., no 'mailHost' nor 'mailRoutingAddress'), the MTA will bounce (reject) the message. There are two exceptions to this rule, to accommodate location-independent accounts, as follows.

If the entry has no routing information, but is a mailing list (i.e., has object class 'mailGroup'), the message is handled locally, i.e., the MTA "receives" messages to the address in question, performs the mail group expansion, and resends to the group members. Thus, a mail group can be configured as "location-independent", meaning that it does not require a particular Messaging Server to perform the mail group expansion.

If the entry has no routing information, but has one or more 'mailForwardingAddress' attributes (see Appendix), it is handled locally, i.e., the MTA "receives" messages to the address in question under the assumption that it is a forwarding-only (or "redirect") account, and forwards the message to the new address(es). Thus, it is not necessary to designate a particular Messaging Server to perform forwarding on behalf of a forwarding-only account. (This exception may be deprecated in a future version, and then all 'nsMessagingServerUser' accounts will require a 'mailHost' value. If location-independent redirects are still desired, a 'mailGroup' entry can be used to achieve the same effect. Or, one could imagine a new object class to combine with 'mailRecipient', say, 'mailForwardingAlias', that just provides a way to configure a location-independent recipient that has a 'mailForwardingAddress', but this may be overkill. One might also consider whether the desired action is actually "routing", not "forwarding" - see Sec. 3.3 for clarification. The point is that a mail server should never perform "forwarding" unless it also takes responsibility for the account's other attributes that specify delivery-time handling, if any; this is to ensure that all of the account's forwarding and delivery preferences are acted upon exactly once in the life of a message.)

Note that if there were a non-Netscape MTA in the environment that implemented the 'mailRecipient' concept but did not mimic the Netscape MTA behavior regarding the above exception cases, it would probably be unadvisable for administrators to configure any accounts

as location-independent. (This suggests that if it is generally useful to configure a certain recipient type as location-independent,

e.g., 'mailGroup', it ought to be standardized.)

4.3 Determining how to Route a Message

If the recipient is not local, but has a 'mailHost' and/or 'mailRoutingAddress' attribute in its LDAP entry, we route the message as follows.

First, we determine a destination. If a 'mailHost' value is present, that is taken to be the destination. Otherwise, the domain part of the 'mailRoutingAddress' value is taken to be the destination.

Second, we determine whether and how to rewrite the SMTP envelope recipient address. If a 'mailRoutingAddress' value is present, the envelope address is rewritten to that. Otherwise, depending on the configuration, the envelope address may be rewritten by combining the 'uid' value, if present, with the 'mailHost' value (e.g., "uid@mail.host"), or, it is rewritten by combining the original envelope address local part with the 'mailHost' value (e.g., "orig.localpart@mail.host"), or it is not rewritten at all.

Third, we determine the next SMTP hop. This may or may not be the same as the destination determined above. Given the destination, the MTA will consult the routing table in the MTA configuration, and/or consult DNS for "MX" and/or "A" records [9].

The message is then relayed to the next SMTP hop, with the SMTP envelope recipient address set as determined above.

Note that if both 'mailHost' and 'mailRoutingAddress' are present, the 'mailHost' attribute determines the destination while the 'mailRoutingAddress' attribute determines the envelope rewrite. It is expected that specifying both is unnecessary, although not inherently harmful, and may be useful in some peculiar cases.

Note also that envelope rewrites may be considered unnecessary (e.g., in Netscape-only MTA sites), and perhaps undesirable (e.g., if the user has multiple addresses and the target MTA allows the user to configure server-side filters that read the envelope; also, envelope

rewrites may increase the chances of "namespace crossovers" in multi-domain sites, as mentioned in Sec. 5.8). Envelope rewrites become necessary when routing to MTAs whose reckoning of their accounts' email addresses is not consistent with the accounts' respective LDAP entries (which could be the case with MTAs that are not 'mailRecipient'-compatible).

5. Examples

The following is a set of directory entries, shown in LDIF [10] format, that illustrates the use of the 'mailRecipient' object class. Examples based on this set of entries are provided in the sections that follow. Each example explains what happens when a message arrives on nsmail1.example.com for the indicated recipient.

```
dn: cn=Joe Blow,o=Example Corp,c=US
objectclass: top
objectclass: person
objectclass: organizationalPerson
objectclass: inetOrgPerson
objectclass: mailRecipient
objectclass: nsMessagingServerUser
cn: Joe Blow
sn: Blow
uid: joeblow
userpassword: {crypt}y9LyrzNBT49Ao
mail: joeblow@example.com
mailhost: nsmail1.example.com
maildeliveryoption: mailbox

dn: cn=John Doe,o=Example Corp,c=US
objectclass: top
objectclass: person
objectclass: organizationalPerson
objectclass: inetOrgPerson
objectclass: mailRecipient
objectclass: nsMessagingServerUser
cn: John Doe
sn: Doe
uid: johndoe
userpassword: {crypt}y9LyrzNBT49Ao
```

```
mail: johndoe@example.com
mailalternateaddress: jonjon@example.com
mailhost: nsmail2.example.com
maildeliveryoption: mailbox

dn: cn=Scuba Group,o=Example Corp,c=US
objectclass: top
objectclass: groupOfUniqueNames
objectclass: mailRecipient
objectclass: mailGroup
cn: Scuba Group
mail: scuba@example.com
mgrprfc822mailmember: joeblow@example.com
mgrprfc822mailmember: johndoe@example.com

dn: cn=Tuba Group,o=Example Corp,c=US
```



```
objectclass: top
objectclass: groupOfUniqueNames
objectclass: mailRecipient
objectclass: mailGroup
cn: Tuba Group
mail: tuba@example.com
mailhost: nsmail2.example.com
mgrprfc822mailmember: joeblow@example.com
mgrprfc822mailmember: janeroe@example.com

dn: cn=Jane Roe,o=Example Corp,c=US
objectclass: top
objectclass: person
objectclass: organizationalPerson
objectclass: inetOrgPerson
objectclass: mailRecipient
objectclass: nsMessagingServerUser
cn: Jane Roe
sn: Doe
uid: janeroe
userpassword: {crypt}y9LyrzNBT49Ao
mail: janeroe@example.com
mailhost: nsmail1.example.com
maildeliveryoption: mailbox
mailforwardingaddress: babs@example.com
```

```
dn: cn=J Random User,o=Example Corp,c=US
objectclass: top
objectclass: mailRecipient
objectclass: nsMessagingServerUser
cn: J Random User
sn: User
mail: jruser@example.com
mailforwardingaddress: random@pu.edu

dn: cn=Babs Jensen,o=Example Corp,c=US
objectclass: top
objectclass: person
objectclass: organizationalPerson
objectclass: inetOrgPerson
objectclass: mailRecipient
objectclass: xyzMailUser
cn: Babs Jensen
sn: Jensen
uid: babs
userpassword: {crypt}y9LyrzNBT49Ao
mail: babs@example.com
mailalternateaddress: bj@schooldist12.k12.ca.us
```



```
mailroutingaddress: Babs_Jensen@xyz1.example.com
xyzPostOfficeName: Example_PO_1
xyzUserType: regular
xyzQuota: 1000000

dn: cn=Charlie Hacker,o=Example Corp,c=US
objectclass: top
objectclass: person
objectclass: organizationalPerson
objectclass: inetOrgPerson
objectclass: mailRecipient
objectclass: nsMessagingServerUser
cn: Charlie Hacker
sn: Hacker
uid: hacker
userpassword: {crypt}y9LyrzNBT49Ao
mail: hacker@schooldist12.k12.ca.us
mailhost: nsmail2.example.com
```

```
mailroutingaddress: hacker@schooldist12.k12.ca.us
maildeliveryoption: mailbox
mailforwardingaddress: babs@example.com

dn: cn=Conference Room 102,o=Example Corp,c=US
objectclass: top
objectclass: conferenceRoom
mail: babs@example.com
roomNumber: 102
```

5.1 Example #1

When a message arrives on nsmail1.example.com for joeblow@example.com, the message is deposited in Joe Blow's mailbox.

5.2 Example #2

When a message arrives on nsmail1.example.com for johndoe@example.com or for jonjon@example.com, the message is relayed to nsmail2.example.com, with "johndoe@nsmail2.example.com" in the envelope (assuming the "uid@mail.host" rewrite option is enabled on nsmail1.example.com). On nsmail2.example.com, the message is identified as belonging to John Doe by virtue of "nsmail2.example.com" being local and "johndoe" being the 'uid' of John Doe (assuming the 'uid' fallback search is enabled on nsmail2.example.com). So the message is deposited in his mailbox on nsmail2.example.com.

The above case would also succeed if the "truncate host part" fallback search were enabled on nsmail2.example.com, or if no

fallback searches or envelope rewrites were configured on either machine (in which case the envelope recipient address would remain unchanged).

5.3 Example #3

When a message arrives on nsmail1.example.com for scuba@example.com, the message is resent to joeblow@example.com and johndoe@example.com. (The message is considered to be locally handlable since the recipient is a mail group with no routing information.)

5.4 Example #4

When a message arrives on nsmail1.example.com for tuba@example.com, the message is relayed to nsmail2.example.com with "tuba@nsmail2.example.com" (assuming that the "orig.localpart@mail.host" option is enabled). On nsmail2.example.com, the message is identified as belonging to the Tuba Group by virtue of the "truncate host part" fallback search, so the message is accepted and resent to the group members.

As in Example #2, the above case would also succeed if no fallback searches or envelope rewrites were configured on either machine.

5.5 Example #5

When a message arrives on nsmail1.example.com for janeroe@example.com, the message is delivered to Jane's mailbox, and is also forwarded to Babs. Perhaps Jane is on leave.

5.6 Example #6

When a message arrives on nsmail1.example.com for jruser@example.com, it is forwarded to random@pu.edu. Perhaps he has left the company to go back to school, and the company is forwarding his mail as a favor.

Note that the presence or absence of the usual object classes such as 'person' do not affect the Messaging Server. Also, the absence of 'uid' and 'userPassword' is probably a good idea since a person who has left the company should not be able to login. Note also that a 'mailHost' could have been specified, e.g., as "nsmail2.example.com", with no difference in overall effect, except that it would require all messages addressed to this user to be passed to nsmail2.example.com where the forward action would then be performed.

(This is an example of a location-independent "redirect" account, which may be deprecated in a future release; see Sec. 4.2.)

5.7 Example #7

When a message arrives on nsmail1.example.com for babs@example.com, or for bj@schooldist12.k12.ca.us (the company is doing a favor to a

local school district by hosting their mail accounts on the company servers; Babs is both an employee in the company and a volunteer at the school district, and so she has both addresses), the message is relayed to the SMTP MTA on host xyz1.example.com (which may be an SMTP-to-XYZ gateway), with "Babs_Jensen@xyz1.example.com" in the envelope.

Note that Conference Room 102 is not identified by the MTA as a recipient of mail addressed to babs@example.com, despite it's having the matching 'mail' address. This is because it does not have the 'mailRecipient' object class.

5.8 Example #8

When a message arrives on nsmail1.example.com for hacker@schooldist12.k12.ca.us, the message is relayed to nsmail2.example.com with "hacker@schooldist12.k12.ca.us" in the envelope. Mail arriving on nsmail2.example.com for this user is deposited into his mailbox, and a copy is forwarded to Babs. Charlie is a guest user from a local school district, and is not in the company, and therefore does not have an address with "example.com".

The reason to force the envelope using 'mailRoutingAddress' is to avoid having it rewritten to "hacker@nsmail2.example.com", which would happen if envelope rewrites using 'mailHost' are enabled. Thus, we avoid a "namespace crossover" that could result in misdelivered mail if there were some other user with address "hacker@example.com". This is one of the peculiar cases where having both 'mailHost' and 'mailRoutingAddress' is useful, since 'mailRoutingAddress' overrides the default rewrite rule (although the problem could also be solved by disabling envelope rewrites, assuming they are not needed). Any site that hosts multiple domains (e.g., an Internet service provider) must be especially careful in considering whether and how envelopes are to be rewritten when mail is routed among its MTAs. (See also Sec. 4.3.)

6. Security Considerations

As in all cases where account information is stored in an LDAP-based directory service, network administrators must be careful to ensure that their directory service controls users' access to the entries and attributes stored therein, according to site policy (e.g., allowing users to modify, say, their 'mailForwardingAddress' attribute, but not their 'mailHost' attribute). Mail server products

and their associated user management tools should help administrators to ensure this, and should also help administrators avoid configurations that would result in misdelivered mail due to "namespace crossovers" and other reasons.

7. Acknowledgements

Many members of the Netscape Messaging Server and Directory Server teams contributed to the design of this schema, including Bill Fitler, Prabhat Keni, Mike Macgirvin, Bruce Steinback, John Myers, Tim Howes, Mark Smith, and John Kristian (who coined the object class name 'mailRecipient'). Special thanks to Leif Hedstrom, Netscape's Chief Dogfood Taster, for his "real world" insights. Thanks also to Jeff Hodges for contributing to the discussion that led to this memo, and to Stuart Freedman for providing review comments.

8. References

[1] W. Yeong, T. Howes, S. Kille, "Lightweight Directory Access Protocol", RFC 1777, March 1995.

[2] "Information Processing Systems - Open Systems Interconnection - The Directory: Overview of Concepts, Models and Service", ISO/IEC JTC 1/SC21, International Standard 9594-1, 1988.

[3] J. Postel, "Simple Mail Transfer Protocol", STD 10, RFC 821, August 1982.

[4] D. Crocker, "Standard for the Format of ARPA Internet Text Messages", STD 11, RFC 822, August 1982.

[5] B. Steinback, "Using LDAP for SMTP Mailing Lists and Aliases", Internet-Draft (work in progress).

[6] M. Wahl, A. Coulbeck, T. Howes, S. Kille, "Lightweight X.500 Directory Access Protocol (v3): Attribute Syntax Definitions", RFC 2252, December 1997.

[7] M. Wahl, "A Summary of the X.500(96) User Schema for use with LDAPv3", RFC 2256, December 1997.

[8] P. Barker, S. Kille, "The COSINE and Internet X.500 Schema", RFC 1274, November 1991.

[9] C. Partridge, "Mail routing and the domain system", STD 14, RFC 974, January 1986.

[10] G. Good, "The LDAP Data Interchange Format (LDIF) - Technical Specification", Internet-Draft (work in progress).

[11] M. Smith, "The inetOrgPerson Object Class", Internet-Draft (work in progress).

9. Author's Address

Hans Lachman
Netscape Communications Corp.
501 East Middlefield Road
Mountain View, CA 94043

Phone: (650) 254-1900
EMail: lachman@netscape.com

10. Appendix - nsMessagingServerUser Object Class and Attributes

The following is an informal description of the 'nsMessagingServerUser' object class and associated attributes. It was designed to be used in combination with the 'mailRecipient' and 'inetOrgPerson' [11] object classes to define a Netscape Messaging Server user account. This definition is not considered part of the 'mailRecipient' definition, and is provided here purely as supplemental information. These attributes may change across releases, and such changes would not affect MTA interoperability.

Object class: nsMessagingServerUser
    Allowed attributes:
        cn
                Common name (person's full name).
        mailAccessDomain
                Domains and IP addresses from which user may do POP or IMAP login.
        mailAutoReplyMode
                Auto-reply mode, may be one (or none) of: 'vacation' (send reply text to sender, but only once per vacation), 'reply' (send reply text unconditionally), or 'echo' (like 'reply' but include original message in the reply).
        mailAutoReplyText

Reply text to use with 'mailAutoReplyMode'.

mailDeliveryOption
: Mail delivery option, one or more of: 'mailbox' (deliver to user's POP/IMAP mailbox), 'native' (deliver with platform's native delivery method, e.g., "/usr/bin/mail"), or 'program' (perform program delivery). There must be at least one 'mailDeliveryOption' and/or 'mailForwardingAddress',

otherwise, mail to this account is undeliverable.

mailForwardingAddress
: User-specifiable mail forwarding address(es). This is different from 'mailRoutingAddress' in that it is intended to be configurable by the user, and may be multi-valued. Thus, forwarding and delivery options may be thought of as "account preferences", while routing attributes are used to get a message to the MTA that will take responsibility for handling the message as per the recipient's account preferences.

mailMessageStore
: Identifier for the message store containing this user's POP/IMAP mailbox.

mailProgramDeliveryInfo
: Command text for program delivery.

mailQuota
: Quota in bytes for user's POP/IMAP mailbox.

11. Full Copyright Statement

MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE.