Secure mail makes it feasible to allow privileged users such as IO.SysDaemon and Backup.SysDaemon to send mail. A new subroutine called send_mail_ sends an ASCII message, with or without an accompanying wakeup, to the ring 1 mailbox used by secure mail (See MTR-064 on secure mail and messages). User programs can call this subroutine, as can system programs that have need to inform people that requests have been performed. Future mail sending and interprocess message sending commands will be based on send_mail_.

A feature of send_mail_ is the ability to send by name alone when no project id is specified. To do this send_mail_ calls the entry mail_table$lookup, which returns a mailbox pathname for a given registered user.

Users are listed voluntarily in the segment mail_table, which has the following format:

```plaintext
dcl mail_table based (mail_table_ptr),
  version fixed bin(1/) aligned,
  current_size fixed bin(1/) aligned,
  area area ((sys_info$max_seg_size));

dcl table_entry based (tablep);
  person char (22),
  alias char (8),
  ds char (12),
  mbx_length fixed bin(17),
  he_length fixed bin(17),
  mbx char (mbx1 refer mbx_length),
  he char (hel refer he_length);
```

- **person** is the user's registered person id.
- **alias** is his registered alias.
- **ds** is a default destination for printing listings to the user.
- **mbx** is the absolute pathname ending in .mbx of a mailbox or link to a mailbox.
The existence of mail_table and its accompanying hash table mail_table.nl is assured at system start up time by an answering service call to the entry mail_table_exists. If necessary, mail_table_exists creates a hash table by a call to hash_make and creates a segment mail_table containing an initialized area. It allocates, for every PNT entry, an entry in mail_table with the following default information:

```
   person = id in PNT
   alias = alias in PNT
   mbx = >udd>default_project>id>id.mbx
   ds = ""
   he = id
```

When the answering service makes certain changes to the PNT, such as adding a new person or changing a default project, it calls the add, delete or update entry in mail_table. Administrative commands also exist to change mail_table information.

Writeups of the new subroutines and commands follow:
**Name:** send_mail_

The send_mail_ subroutine sends one piece of mail to one user.

**Usage**

dcl send_mail_ entry (char(*), char(*), char(*),
ptr, fixed bin(35));

call send_mail_ (name, project, message, s_ptr, code);

1) name is the recipient's person_id.
2) project is the recipient's project_id. (See Notes)
3) message is the ASCII message.
4) s_ptr is a pointer to the following structure, to comprise the beginning of the added message:

   dcl 1 mail_structure aligned,
       2 version fixed bin(17), /* = 1 */
       2 sent_from char(32) aligned,
       2 lines fixed bin(17),
       2 switches,
       3 wakeup bit(1) unaligned,
       3 urgent bit(1) unaligned,
       3 has_been_read bit(1) unaligned,
       3 acknowledge bit(1) unaligned,
       3 pad bit(32) unaligned init("0"b);

5) code is a standard status code, for example:

   error_table_Suser_not_found
   Not found in mail_table.
   error_table_Snoentry
   Mailbox does not exist.
   error_table_Sno_info
   No information can be returned about recipient.

**Notes**

The mailbox pathname is >udd>project>name>name.mbx.

When project is "", the pathname of a default mailbox is obtained by calling mail_table_Slookup.

Information on how to send is in mail_structure and is stored with the message so that it can be read. The wakeup bit, for example, tells send_mail_ when to send a wakeup and tells the mail and message reading commands which messages to print.

Pad in mail_structure should always be zero to allow for defining new bits.
**Name:** mail_table

This module manages the data base mail_table, which contains information useful for sending mail and printing listings to users.

**Entry:** exists

This entry checks whether mail_table and mail_table.ht exist and if not, creates them. Sexists is usually called by the answering service.

**Usage**

dcl mail_table_Sexists entry
   (ptr, bit(*), fixed bin(35));

call mail_table_Sexists (ansp, switches, segname, code);

1) `ansp` is a pointer to the answer table. (Input)

2) `switches` (Output):

   - `made_one` mail_table and mail_table.ht were created.
   - `old_one` there was an old mail_table. This switch is on only when `made_one` is on.
   - `new_one` a copy of mail_table exists because there is no access to write the original.
   - `fatal_error` a fatal error is reflected in code. A fatal error is something like the PNT being the wrong version.

3) `segname` is the name of the segment referred to by code if code is non-zero and `new_one` is off. If `new_one` is on, `segname` is the name of the copy. (Output)

4) `code` is a normal status code.
Entry: lookup

This entry returns information from mail_table given a user's registered person id or alias as a lookup name.

Usage

dcl mail_table$lookup entry
  (char(35), ptr, fixed bin(35));

call mail_table$lookup (name, argp, code);

1) name is a registered person id or a registered alias. (Input)

2) argp is a pointer to the following structure in which information is returned:

dcl | args based (argp),
  2 version fixed bin(17),
  2 person char (22),
  2 alias char (8),
  2 mbx char (168),
  2 ds char (12),
  2 he char (44);

Every version mail_table has a different args structure. (Input)

3) code is a standard status code, usually either zero or error_table$user_not_found. (Output)

Entry: add

This entry adds a user to mail_table.

Usage

dcl mail_table$add entry (ptr, fixed bin(35));

call mail_table$add (argp, code);

1) argp is a pointer to the above args structure. (Input)

2) code is a standard status code, error_table$nameDup indicates that an entry for person already exists and the caller must use the delete or update entry. (Output)
**Entry: delete**

This entry deletes a user from mail_table.

**Usage**

dcl mail_table_delete entry (char(*), code);
call mail_table_delete (name, code);

1) **name** is a registered person id or alias. (Input)
2) **code** is a standard status code. (Output)

**Entry: update**

This entry updates selective information in mail_table for a particular user.

**Usage**

dcl mail_table_update entry
    (char(*), ptr, fixed bin(35));
call mail_table_update (name, argp, code);

1) **name** is a registered person id or alias. (Input)
2) **argp** is a pointer to the args structure. (Input)
3) **code** is a standard status code. (Output)

**Notes**

If a field in the args structure is "", that field is not updated in mail_table.
mail_table_lookup, mtl

The mail_table_lookup command returns information from mail_table given a user's registered person id or alias.

Usage
mail_table_lookup name -control_arg-

1) name is a registered person id or alias.

Control Argument
-all print the following information:

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Person</td>
<td>registered person id</td>
</tr>
<tr>
<td>Alias</td>
<td>registered alias</td>
</tr>
<tr>
<td>Mailbox</td>
<td>pathname of default mailbox</td>
</tr>
<tr>
<td>Destination</td>
<td>default dprint destination</td>
</tr>
<tr>
<td>Header</td>
<td>default dprint header</td>
</tr>
</tbody>
</table>

Default is to print only Mailbox.

mail_table_add, mta

The administrative command mail_table_add adds a user to mail_table.

Usage
mail_table_add person alias -control_arg-

1) person is a registered person id, maximum of 22 characters.
2) alias is the registered alias, maximum of 8 characters.

Control Arguments

-mailbox path path is the pathname of a default mailbox. If the suffix ".mbx" is not present, it is assumed.
-destination string string is a default destination for the dprint command, no longer than 12 characters.
-header string string is a default header for the dprint command, no longer than 64 characters.

Note
Any fields not specified by control arguments are set to "".
Name: mail_table_delete, mtd

The administrative command mail_table_delete removes a user from mail_table.

Usage
mail_table_delete name

1) name is a registered person id or alias.

Note
An error message is printed if no entry for name exists.

Name: mail_table_update, mtu

The administrative command mail_table_update updates information in mail_table for a particular user.

Usage
mail_table_update name -control args-

1) name is a registered person id or alias.

Control Arguments

-mailbox path
- mbx path
path is the path name of a default mailbox. If the suffix " .mbx" is not present, it is assumed.

-destination string
-ds string
string is a default destination for the dprint command, no longer than 12 characters.

-header string
-he string
string is a default header for the dprint command, no longer than 64 characters.

Notes
Any fields not specified by control arguments are not updated.
This command cannot replace the person id or alias.