RIPEM INTERNET KEY SERVER
User's Guide

Gary Ratterree
Texas A&M University

garyr@cs.tamu.edu

March 1993

Portions of this manual have been adapted from the RIPEM User's Guide (December 1992 edition) by Mark Riordan mrr@scss3.cl.msu.edu

I. Using the RIPEM Key Server:

Key Server Description and Limitations

If you have Internet access, you can communicate your key to others by registering the key on an Internet RIPEM key server. Currently, there is an "experimental" RIPEM key server running on the host dsls4.cs.tamu.edu. This host is experimental in that it is an unofficial service which may have to be terminated with little or no advance notice.

This RIPEM key server acts as a central repository for public keys, saving users the effort of distributing their keys individually to all potential correspondents. This key server is not an especially secure mechanism. The level of security present in the key protocols is much less than that provided, for instance, by the Privacy Enhanced Mail certificate mechanism specified in the Internet PEM RFC's. The authenticity of keys maintained on the server is not guaranteed. The RIPEM key server is simply a means for RIPEM users to conveniently exchange keys.

Registering a Key via the Key Server

To allow the maximum number of users to publicize their keys via this mechanism, the RIPEM key server accepts key registration requests by electronic mail. Although the RIPEM key server itself is connected only to the Internet, users of non-Internet networks such as CompuServe, BITNET, and so on can register their keys by sending their key registration requests via an appropriate network gateway.

To register your key, send the public component (the output file from the -P option) to the email address

```
ripem-register-keys@cs.tamu.edu
```

On a UNIX system, for instance, you can register your key by a command like:

```
mail ripem-register-keys@cs.tamu.edu <mypublickey
```

The key server will register your public key in its database and will send you a confirming message. The key is identified by the email address specified during the generation of the key, but the confirming message is sent to the address from which the key file was sent.

If you read electronic mail on several different hosts but wish to use the same public key on all of them, you can register the key under multiple names. You can do this by editing the key file before sending it to the server, and adding additional `User:` lines. (See the separate document on file formats for more information.)

```
-----BEGIN PUBLIC KEY-----
User: garyr@cs.tamu.edu
User: garyr@cs
User: garyr
PublicKeyInfo:
 MFgwCgYEVQgBAQICAf8DSgAwRwJAbiMDTw/j/q7H7hUDBFYV/+8NBBJM/IsaIY1G
 s6SByD1NRBqCkblUq4iYETsW0/JQKieZYwYntNAgu3+bRIuUxwIDAQAB
MD5OfPublicKey: 5A677077AB7C6D9DD358224A72735CC7
-----END PUBLIC KEY-----
```

Or, you can register the key under different names via separate email messages.

To subsequently delete your key from the server, encrypt a message starting with the string

```
RemoveKey
```

with the `-m mic-only` command line option and send the encrypted message to the address:

```
ripem-remove-keys@cs.tamu.edu
```

The message must have been encrypted by the owner of the key being removed.

To change your key on the server, generate a new key pair and encrypt the public component (the file from RIPEM's `-P` option) with the `-m mic-only` command line option. Send the result to the address:

```
ripem-change-keys@cs.tamu.edu
```

The message must have been encrypted by the owner of the key being changed.

Obtaining Keys from the Key Server: Live Access

Real-time "live" queries to the RIPEM key server are made directly by RIPEM using the UDP IP network protocol. "Live" queries are possible if your computer is connected to the Internet, your copy of RIPEM has been compiled for network access, and your computer is running the right network software. This is often true of UNIX computers but is generally not true of other computers. At this writing, for instance, the MS-DOS version of RIPEM supports only the PC/TCP network software from FTP Software, Inc.

In order to access the key server, RIPEM needs to know its Internet address. You can tell RIPEM the address of the server in two ways: you can set the environment variable `RIPEM_SERVER_NAME` to the name of the server, or you can specify the server name with the `-y` command line option. In either case, you can specify more than one server name, separating the server names with commas (and no blank spaces). If you specify a list of server names in this way, when querying servers RIPEM will query the servers in the order listed until it obtains the desired public key, or exhausts the list.

Obtaining Keys from the Key Server: Email-Only Access

For users for whom live UDP network access to the RIPEM key server is not possible or not feasible, electronic mail access to the key server has been implemented. To obtain a copy of the complete database of registered RIPEM keys via email, send a message to the address:

```
ripem-get-keys@cs.tamu.edu
```

The subject and content of the message are ignored by the server, and hence can be left blank.

The return email message will contain a flat file of public keys, readable directly by RIPEM. This file can be used as described in the section discussing flat files in the RIPEM User's Guide.

II. KEY SERVER PROGRAMS:

copymail:

The copymail program simply copies incoming messages to separate files into the correct "incoming" directory. This program receives public key addition, change, and deletion requests via e-mail over the Internet. This program is run on demand (i.e. not in the background) when ever mail arrives. Section III describes how to create the correct mail aliases entries for copymail.

copymail has the following command line options:

- -d (directory) This switch tells copymail which directory to place incoming mail in.

givepubs:

The givepubs program responds to email requests for the flat file version of the database by mailing back the file to the requester. This program provides the service of distributing the RIPEM public key database by email.

givepubs has the following command line options:

- -f (filename) This is the name of the file that givepubs is to return.
- -s (subject) This string will appear on the subject line of the return message.

rkeyreg:

Rkeyreg is the key registration program. It runs in the background, periodically waking up to scan copymail's "incoming" directories. When a new file has arrived, rkeyreg sends a signal to rkeyserv, asking it to grant access to the real, random-access key database. rkeyreg then reads through the incoming files, extracting keys and performing the requested operations on the key database. It then sends a message to the sender of each file (not necessarily the owner of the keys in the file), reporting on the success or failure of the attempt to register the key(s).

rkeyreg has the following command line options:

- -a (add directory) The directory to scan for incoming key addition requests.
- -r (remove directory) The directory to scan for key removal requests.
- -c (change directory) The directory to scan for key change requests.
- -t (timeout) The number of seconds we wait before each access to the key database. This keeps us from hogging the database and allows UDP requests to be processed in a timely manner. Default value is 3 seconds.
- -l (log file) The name of a file to keep a running request log in.
- -s (timeout scan) The number of seconds to wait between scans of the incoming file directories. Larger values create slower responses, but less load on the CPU. Default value is 10 seconds.
- -o The name of the key database file (the one in GDBM format)

-p The name of the file where rkeyerv's process id is contained. This argument allows rkeyreg to communicate with the server for access to the database.
-D (debug) Turns on debugging output when not zero. Useful for error correction. Values of 1,2, and 3 turn on various error and information messages.
-Z (debug file) Name of a file to redirect debugging output to.

rkeyserv:

The main server program is rkeyserv. This program runs in the background, waiting for UDP requests for key lookup from RIPEM programs. It responds by looking up the key in the database and sending back a UDP packet containing the key. The server is a stateless UDP-based server. It normally has the key database open read-only, but closes the database upon request by rkeyreg.

rkeyserv has the following command line options:

-g (database name) The name of the file to use as the key database. Must be in GDBM 1.5 format.
-p (port number) The number of the UDP port to use. Default value is 1611.
-f (fast) Turns off unnecessary system calls.
-P (process ID file) The name of the file to store the rkeyserv's process id into. This file is used by rkeyreg to coordinate database access.
-t (timeout) The number of seconds to wait on the UDP socket select before timing out. Default value is one second.
-d (detach) Tells rkeyserv not to detach itself from the controlling terminal.
-D (debug) Turns on debugging output if greater than zero. Default is zero.
-Z (debug file) The name of a file to redirect debugging output to.

db2flat:

This program converts the key database (which is kept in GDBM format) to a flat file that can be read directly by RIPEM. This program is usually run periodically as a cron job to create the key file that is distributed via email.

db2flat has the following command line options:

-g (database name) The name of the GDBM 1.5 format file to convert.
-o (output file) The name of the file to create the ASCII flat file in.
-t (timeout) Not used???
-D (debug) Turns on debugging output if greater than one.
-Z (debug file) The name of a file to redirect debugging output to.

crekeydb:

This program coverts an ASCII flat file that can be read by RIPEM into the key database format (which is kept in GDBM format). This program is useful for updating the local database from flat files obtained from other key servers.

crekeydb has the following command line options:

-p (flatfile) The name of the ASCII flat file to convert.
Default value is testpub.
-o (database name) The name of the GDBM database to create.
Default value is testkey.gdbm.
-f (fast) Eliminates extra system calls. Unused in this version?
-n (number of items) Number of items to put in the database.
Default value is 100.
-D (debug) Turns on debugging output when not zero.

III. RUNNING THE KEY SERVER:

There are three parts to the RIPEM key server. The first part uses the copymail program to respond to incoming e-mail request to add, change, and delete keys. The rkeyreg program performs the actual database updates for this part of the server. The second part uses the give pubs program to respond to incoming e-mail requests for a copy of the key flat file. The db2flat program is run as a cron job to create the flat file that is sent out. The last part of the key server is rkeyserv. This program runs in the background and responds to UDP request from RIPEM programs for key lookups.

Setup for key addition, changes, and deletion:

To setup the copymail program, you must add entries in the `/etc/aliases` file to redirect incoming mail to the RIPEM key server. The sendmail program will then pipe the incoming mail to the copymail program on demand (i.e. when mail arrives).

The following entries should be added to /etc/aliases (Modified of course for your path and directory names):

```
ripem-register-keys:"|/pub/advos/ripem/server/copymail -d
                      /pub/advos/ripem/server/add-keys"
ripem-remove-keys:  "|/pub/advos/ripem/server/copymail -d
                      /pub/advos/ripem/server/add-keys"
ripem-change-keys:  "|/pub/advos/ripem/server/copymail -d
                      /pub/advos/ripem/server/add-keys"
```

Caution: The current version of SunOS sends all mail alias strings to the shell in lower case. Uppercase filenames and directory entries may be lost.

The rkeyreg program then must be run on demand to process the incoming key registration requests. The following is an example of how this can be done:

```
rkeyreg     -a /pub/advos/ripem/server/add-keys
            -r /pub/advos/ripem/server/remove-keys
            -c /pub/advos/ripem/server/change-keys
            -p /pub/advos/ripem/server/server.pid
            -o /pub/advos/ripem/server/pubkeys.gdbm
            -l /pub/advos/ripem/server/server.log&
```

Setup for requests for the key flat file:

The givepubs program is called to respond to incoming requests via e-mail for the key server flatfile database. The following entry must be added to the file `/etc/aliases` to call givepubs on demand:

```
ripem-get-keys:"|/pub/advos/ripem/server/givepubs
                -f /pub/advos/ripem/server/add-keys
                -s "Your RIPEM KEY REQUEST""
```

In addition, the db2flat program must be run periodically to update the flatfile from the GDBM database file. The following script can be called as a cron job to do this updating:

```
cd /pub/advos/ripem/server
mv pubkeys.txt pubkeys.old
db2flat -g pubkeys.gdbm -o pubkeys.txt
```

Setup for the UDP server:

The UDP server program rkeyserv is run in the background to process incoming UDP requests from RIPEM programs. It should be run on a computer with Internet access (you do not have to use the & since the program will detach itself from the console). The `server.pid` file must be readable by the rkeyreg program. The following is an example of how to start a typical key server session:

```
rkeyserv -g /pub/advos/ripem/server/pubkeys.gdbm -P server.pid
```