

**NAME**

rmtinit, rmtdebug, rmthostname, rmtfilename, rmtgetconn - initiate a connection to a remote tape server  
(-lrmt)

**SYNOPSIS**

```
#include <schily/librmt.h>
```

```
void rmtinit(errmsgn, eexit)
```

```
    int (*errmsgn) (int, const char *, ...);
```

```
    void (*eexit) (int);
```

```
int rmtdebug(dlevel)
```

```
    int dlevel;
```

```
const char *rmtrmt(rmtname)
```

```
    const char *rmtname;
```

```
const char *rmtrsh(rshname)
```

```
    const char *rshname;
```

```
char *rmtfilename(name)
```

```
    char *name;
```

```
char *rmthostname(hostname, hnsz, rmtspec)
```

```
    char *hostname;
```

```
    int hnsz;
```

```
    char *rmtspec;
```

```
int rmtgetconn(host, trsize, excode)
```

```
    char *host;
```

```
    int trsize;
```

```
    int excode;
```

**DESCRIPTION****rmtinit()**

Is an optional function that allows one to set up a function for error printing and a function to be called to exit the program. If **rmtinit(3)** is not called or any of the function pointers is NULL, the appropriate default function is used instead. The supplied error printing function needs to implement the same interface as **errmsgno(3)** and the supplied exit function needs to implement the same interface as **exit(3)**.

**rmtdebug()**

allows one to set the debug level for the library code. The default debug level is 0 and does not print debug messages.

**rmtrmt()**

allows one to set up a different default path to the remote **rmt** server program. The default is otherwise **/etc/rmt**. The **RMT** environment still overwrites the default set up by **rmtrmt(3)**.

**rmtrsh()**

allows one to set up a different remote login program to the remote **rmt** server program. The default is otherwise to use **rcmd(3)**. The **RSH** environment still overwrites the default set up by **rmtrsh(3)**.

**rmtfilename()**

is given a filename that may be either in remote file syntax ( *hostname:filename* or *user@hostname:filename* ) or specified as a hierarchical file path. If the argument turns out to be in remote file syntax, a pointer to the filename part is returned.

**rmthostname()**

This function copies the user/host part of *rmtspec* which should be in remote file syntax. The first argument is a character array that should be large enough to hold the user/host part of *rmtspec*. The second argument is the size of the character array. The third argument is a string in remote file syntax.

**rmtgetconn()**

This function establishes a connection to the remote tape server process. The first parameter is the user/host part of a string in remote file syntax and should be created via **rmthostname(3)**. The second parameter is the expected maximum transfer size. It is used to set up kernel buffering via **setsockopt(3)** to increase performance. The third parameter is an alternate exit code to be used instead of the library default if **rmtgetconn(3)** decides to call **exit(3)**. This allows commands like **ufsdump** to use the documented exit codes for startup errors.

**RETURNS****rmtdebug()**

returns the old debug level.

**rmtrmt()**

returns the old default remote **rmt** server program path or **NULL** in case that the default was not overwritten.

**rmttrsh()**

returns the old default **rmt** remote login program or **NULL** in case that the default was not overwritten.

**rmtfilename()**

returns the filename part of the argument string or **NULL** in case the argument turns out to be not in remote file syntax.

**rmthostname()**

returns a pointer to the first argument or **NULL** in case the *rmtspec* argument turns out to be not in remote file syntax.

**rmtgetconn()**

return a file descriptor which is suitable to be used as first argument for functions like **rmtopen(3)** or **rmtwrite(3)**. If **rmtgetconn(3)** fails to set up a connection, -1 is returned. If **rmtgetconn(3)** is unable to find the port number for shell/tcp, the current uid has no entry in the passwd file or the user name includes illegal characters, **exit()** is called. If you do not like **rmtgetconn(3)** to exit in this case, call **rmtinit(3)** before and install a non exiting function as **exit(3)** handler; **rmtgetconn(3)** then will return -2 after this function did return.

**EXAMPLES**

```

int    remfd;
char   *remfn;
char   host[256];
int    iosize = 10240; /* socket send/receive size to set up */

if ((remfn = rmtfilename(filename)) != NULL) {
    rmthostname(host, sizeof (host), filename);

    if ((remfd = rmtgetconn(host, iosize, 0)) < 0)
        comerrno(EX_BAD, "Cannot get connection to '%s'.\n",
                /* errno not valid !! */      host);
}

if (rmtopen(remfd, remfn, mode) < 0)
    comerr("Cannot open '%s'.\n", remfn);

if (rmtread(remfd, buf, sizeof(buf)) < 0)
    comerr("Read error on '%s'.\n", remfn);

```

**rmtclose(remfd);**

## ENVIRONMENT

### RSH

If the **RSH** environment is present, the remote connection will not be created via **rcmd(3)** but by calling the program pointed to by **RSH**. Use e.g. **RSH=/usr/bin/ssh** to create a secure shell connection.

If the environment **RSH** is empty, then the default **rcmd(3)** is used even in case **rmtrsh(3)** has been called.

### RMT

If the **RMT** environment is present, the remote tape server will not be the program **/etc/rmt** but the program pointed to by **RMT**.

If the environment **RMT** is empty, then the default **/etc/rmt** is used even in case **rmtrmt(3)** has been called.

Note that the remote tape server program name will be ignored if you log in using an account that has been created with a remote tape server program as login shell.

## SEE ALSO

**rmt(1)**, **rsh(1)**, **ssh(1)**, **rcmd(3)**, **rmtinit(3)**, **rmtdebug(3)**, **rmtrmt(3)**, **rmtrsh(3)**, **rmthostname(3)**, **rmtfilename(3)**, **rmtgetconn(3)**, **rmtopen(3)**, **rmtioctl(3)**, **rmtclose(3)**, **rmtread(3)**, **rmtwrite(3)**, **rmtseek(3)**, **rmtxstatus(3)**, **rmtstatus(3)**, **\_mtg2rmtg(3)**, **\_rmtg2mtg(3)**, **errmsgno(3)**

## BUGS

For now (late 2002), we know that the following programs are broken and do not implement signal handling correctly:

rsh on SunOS-5.0...SunOS-5.9

ssh from ssh.com

ssh from openssh.org

Sun already did accept a bug report for **rsh(1)**. Openssh.org accepted a bug for their implementation of **ssh(1)**.

If you use **rmtgetconn**(3) to create a remote connection via an unfixed **rsh**(1) or **ssh**(1), be prepared that terminal generated signals may interrupt the remote connection.

Mail other bugs and suggestions to **schilytools@mlists.in-berlin.de** or open a ticket at **<https://codeberg.org/schilytools/schilytools/issues>**.

The mailing list archive may be found at:

**<https://mlists.in-berlin.de/mailman/listinfo/schilytools-mlists.in-berlin.de>**.

## AUTHORS

**librmt** has been written in 1990 by Joerg Schilling. In 1995, support for **RMT VERSION 1** has been added. **librmt** is now maintained by the schilytools project authors.

## SOURCE DOWNLOAD

The source code for **librmt** is included in the **schilytools** project and may be retrieved from the **schilytools** project at Codeberg at

**<https://codeberg.org/schilytools/schilytools>**.

The download directory is

**<https://codeberg.org/schilytools/schilytools/releases>**.