NAME

libcapsicum — library interface to capability-mode services

LIBRARY

library “libcapsicum”

SYNOPSIS

```
#include <sys/types.h>
#include <sys/capability.h>
#include <libcapsicum.h>

int
lcs_get(struct lc_host **lchpp);

int
lcs_getsock(struct lc_host *lchp, int *fdp);

ssize_t
lcs_recv(struct lc_host *lchp, void *buf, size_t len, int flags);

ssize_t
lcs_recv_rights(struct lc_host *lchp, void *buf, size_t len, int flags,
    int *fdp, int *fdcountp);

int
lcs_recvrpc(struct lc_host *lchp, u_int32_t *opnop, u_int32_t *seqnop,
    u_char **bufferp, size_t *lenp);

int
lcs_recvrpc_rights(struct lc_host *lchp, u_int32_t *opnop,
    u_int32_t *seqnop, u_char **bufferp, size_t *lenp, int *fdp,
    int *fdcountp);

ssize_t
lcs_send(struct lc_host *lchp, const void *msg, size_t len, int flags);

ssize_t
lcs_send_rights(struct lc_host *lchp, const void *msg, size_t len,
    int flags, int *fdp, int fdcount);

int
lcs_sendrpc(struct lc_host *lchp, u_int32_t opno, u_int32_t seqno,
    struct iovec *rep, int repcount);

int
lcs_sendrpc_rights(struct lc_host *lchp, u_int32_t opno, u_int32_t seqno,
    struct iovec *rep, int repcount, int *fdp, int fdcount);
```

DESCRIPTION

The **libcapsicum** library routines provide services for processes hosting or running in capability mode. Depending on the requirements of the host and sandbox, the API can simply be used to set up and stop sandboxes, used to manage I/O using a `unix`(4) domain socket connection to the sandbox, or can provide a basic remote procedure call (RPC) facility. Applications may also use RPC generators such as `rpcgen`(1) to build event handling and marshaling code.

This man page describes the sandbox API. General information on **libcapsicum** may be found in `libcapsicum`(3).

SANDBOX API

The **libcapsicum** sandbox API allows sandbox processes to interact with their host process. Sandbox API functions can be identified by their function name prefix, `lcs_`.

Each executing sandbox will have a single corresponding host instance, described by an opaque which is returned by **lcs_get**().

The socket for the host may be queried using **lcs_getsock**().

libcapsicum implements a number of I/O functions as part of the sandbox API, which are documented in libcapsicum_sandbox(3). **lcs_recv**() and **lcs_send**() provide simple wrappers around recv(2) and send(2) to avoid sandboxes having to query host socket file descriptors before use.

lcs_recv_rights() and **lcs_send_rights**() similarly allow receiving and sending file descriptors with messages.

lcs_recvrpc() and **lcs_sendrpc**() may be used to implement a simple RPC system, in coordination with a host using **lch_rpc**(). **lcs_recvrpc**() blocks awaiting the receipt of an RPC request, which will be returned in a buffer allocated using malloc(3), *bufferp*, and with a data size returned via *lenp*. The caller will also receive an operation number and a sequence number via *opnop* and *seqnop*.

When an RPC is complete, it should be returned to the host via **lcs_sendrpc**(), which accepts the same operation and sequence number as arguments, as well as reply data via the *iovec rep* and *repcount*. When the sandbox is done with the request data, it should free the memory using free(3). **lcs_recvrpc_rights** and **lcs_sendrpc_rights** allow the receiving and sending of file descriptors along with the RPC.

SEE ALSO

rpcgen(1), recv(2), send(2), writev(2), free(3), libcapsicum(3), libcapsicum_host(3), malloc(3), unix(4)

HISTORY

Support for capabilities and capabilities mode was developed as part of the TrustedBSD Project.

BUGS

WARNING: THIS IS EXPERIMENTAL SECURITY SOFTWARE THAT MUST NOT BE RELIED ON IN PRODUCTION SYSTEMS. IT WILL BREAK YOUR SOFTWARE IN NEW AND UNEXPECTED WAYS.

AUTHORS

These functions and the capability facility were created by Robert N. M. Watson at the University of Cambridge Computer Laboratory with support from a grant from Google, Inc.