CLD

0.1git

Generated by Doxygen 1.8.7

Sun Aug 17 2014 01:53:27

Contents

Chapter 1

Data Structure Index

1.1 Data Structures

Here are the data structures with brief descriptions:

Chapter 2

File Index

2.1 File List

Here is a list of all files with brief descriptions:

Chapter 3

Data Structure Documentation

3.1 chunk_check_status Struct Reference

```
#include <chunk_msg.h>
```

Data Fields

- uint8_t state
- uint8_t pad [3]
- uint32_t count
- uint64_t lastdone

3.1.1 Field Documentation

3.1.1.1 uint32_t chunk_check_status::count

3.1.1.2 uint64_t chunk_check_status::lastdone

3.1.1.3 uint8_t chunk_check_status::pad[3]

3.1.1.4 uint8_t chunk_check_status::state

The documentation for this struct was generated from the following file:

- include/chunk_msg.h

3.2 chunksrv_req Struct Reference

```
#include <chunk_msg.h>
```

Data Fields

- uint8_t magic [CHD_MAGIC_SZ]
- uint8_t op
- uint8_t flags
- uint16_t key_len
- uint32_t nonce

- uint64_t data_len
- char sig [CHD_SIG_SZ]

3.2.1 Field Documentation

3.2.1.1 uint64_t chunksrv_req::data_len

3.2.1.2 uint8_t chunksrv_req::flags

3.2.1.3 uint16_t chunksrv_req::key_len

3.2.1.4 uint8_t chunksrv_req::magic[CHD_MAGIC_SZ]

3.2.1.5 uint32_t chunksrv_req::nonce

3.2.1.6 uint8_t chunksrv_req::op

3.2.1.7 char chunksrv_req::sig[CHD_SIG_SZ]

The documentation for this struct was generated from the following file:

- include/chunk_msg.h

3.3 chunksrv_resp Struct Reference

```
#include <chunk_msg.h>
```

Data Fields

- uint8_t magic [CHD_MAGIC_SZ]
- uint8_t resp_code
- uint8_t rsv1 [3]
- uint32_t nonce
- uint64_t data_len
- unsigned char hash [CHD_CSUM_SZ]

3.3.1 Field Documentation

3.3.1.1 uint64_t chunksrv_resp::data_len

3.3.1.2 unsigned char chunksrv_resp::hash[CHD_CSUM_SZ]

3.3.1.3 uint8_t chunksrv_resp::magic[CHD_MAGIC_SZ]

3.3.1.4 uint32_t chunksrv_resp::nonce

3.3.1.5 uint8_t chunksrv_resp::resp_code

3.3.1.6 uint8_t chunksrv_resp::rsv1[3]

The documentation for this struct was generated from the following file:

- include/chunk_msg.h

3.4 chunksrv_resp_chkstat Struct Reference

```
#include <chunk_msg.h>
```

Data Fields

- struct chunksrv_resp resp
- struct chunk_check_status chkstat

3.4.1 Field Documentation

3.4.1.1 struct **chunk_check_status** chunksrv_resp_chkstat::chkstat

3.4.1.2 struct **chunksrv_resp** chunksrv_resp_chkstat::resp

The documentation for this struct was generated from the following file:

- include/chunk_msg.h

3.5 chunksrv_resp_get Struct Reference

```
#include <chunk_msg.h>
```

Data Fields

- struct chunksrv_resp resp
- uint64_t mtime

3.5.1 Field Documentation

3.5.1.1 uint64_t chunksrv_resp_get::mtime

3.5.1.2 struct **chunksrv_resp** chunksrv_resp_get::resp

The documentation for this struct was generated from the following file:

- include/chunk_msg.h

3.6 cld_dirent_cur Struct Reference

```
#include <cldc.h>
```

Data Fields

- const void ∗ p
- size_t tmp_len

3.6.1 Field Documentation

3.6.1.1 const void∗ cld_dirent_cur::p

3.6.1.2 size_t cld_dirent_cur::tmp_len

The documentation for this struct was generated from the following file:

- include/cldc.h

3.7 cld_timer Struct Reference

```
#include <cld_common.h>
```

Data Fields

- bool fired
- bool on_list
- void(∗ cb)(struct cld_timer ∗)
- void ∗ userdata
- time_t expires
- char name [32]

3.7.1 Field Documentation

3.7.1.1 void(∗ cld_timer::cb)(struct **cld_timer** ∗)

3.7.1.2 time_t cld_timer::expires

3.7.1.3 bool cld_timer::fired

3.7.1.4 char cld_timer::name[32]

3.7.1.5 bool cld_timer::on_list

3.7.1.6 void∗ cld_timer::userdata

The documentation for this struct was generated from the following file:

- include/cld_common.h

3.8 cld_timer_list Struct Reference

```
#include <cld_common.h>
```

Data Fields

- GList ∗ list
- time_t runmark

3.8.1 Field Documentation

3.8.1.1 GList∗ cld_timer_list::list

3.8.1.2 time_t cld_timer_list::runmark

The documentation for this struct was generated from the following file:

- include/cld_common.h

3.9 cldc_call_opts Struct Reference

per-operation application options

```
#include <cldc.h>
```

Data Fields

- int(∗ cb)(struct cldc_call_opts ∗, enum cle_err_codes)
- void ∗ private
- struct cld_msg_get_resp resp

3.9.1 Detailed Description

per-operation application options

3.9.2 Field Documentation

3.9.2.1 int(∗ cldc_call_opts::cb)(struct cldc_call_opts ∗, enum cle_err_codes)

3.9.2.2 void∗ cldc_call_opts::private

3.9.2.3 struct cld_msg_get_resp cldc_call_opts::resp

The documentation for this struct was generated from the following file:

- include/cldc.h

3.10 cldc_fh Struct Reference

an open file handle associated with a session

```
#include <cldc.h>
```

Data Fields

- uint64_t fh
- struct cldc_session ∗ sess
- bool valid

3.10.1 Detailed Description

an open file handle associated with a session

3.10.2 Field Documentation

3.10.2.1 uint64_t cldc_fh::fh

3.10.2.2 struct **cldc_session**∗ cldc_fh::sess

3.10.2.3 bool cldc_fh::valid

The documentation for this struct was generated from the following file:

- include/cldc.h

3.11 cldc_host Struct Reference

Information for a single CLD server host.

```
#include <cldc.h>
```

Data Fields

- unsigned int prio
- unsigned int weight
- char ∗ host
- unsigned short port

3.11.1 Detailed Description

Information for a single CLD server host.

3.11.2 Field Documentation

3.11.2.1 char∗ cldc_host::host

3.11.2.2 unsigned short cldc_host::port

3.11.2.3 unsigned int cldc_host::prio

3.11.2.4 unsigned int cldc_host::weight

The documentation for this struct was generated from the following file:

- include/cldc.h

3.12 cldc_msg Struct Reference

an outgoing message, from client to server

```
#include <cldc.h>
```

Data Fields

- uint64_t xid
- enum cld_msg_op op
- struct cldc_session ∗ sess
- ssize_t(∗ cb)(struct cldc_msg ∗, const void ∗, size_t, enum cle_err_codes)
- void ∗ cb_private
- struct cldc_call_opts copts
- bool done
- time_t expire_time
- int n_pkts
- struct cldc_pkt_info ∗ pkt_info [0]

3.12.1 Detailed Description

an outgoing message, from client to server

3.12.2 Field Documentation

3.12.2.1 ssize_t(∗ cldc_msg::cb)(struct cldc_msg ∗, const void ∗, size_t, enum cle_err_codes)

3.12.2.2 void∗ cldc_msg::cb_private

3.12.2.3 struct cldc_call_opts cldc_msg::copts

3.12.2.4 bool cldc_msg::done

3.12.2.5 time_t cldc_msg::expire_time

3.12.2.6 int cldc_msg::n_pkts

3.12.2.7 enum cld_msg_op cldc_msg::op

3.12.2.8 struct cldc_pkt_info∗ cldc_msg::pkt_info[0]

3.12.2.9 struct cldc_session∗ cldc_msg::sess

3.12.2.10 uint64_t cldc_msg::xid

The documentation for this struct was generated from the following file:

- include/cldc.h

3.13 cldc_node_metadata Struct Reference

```
#include <cldc.h>
```

Data Fields

- quad_t inum
- quad_t vers
- quad_t time_create

- quad_t time_modify
- int flags
- const char ∗ inode_name

3.13.1 Field Documentation

3.13.1.1 int cldc_node_metadata::flags

3.13.1.2 const char∗ cldc_node_metadata::inode_name

3.13.1.3 quad_t cldc_node_metadata::inum

3.13.1.4 quad_t cldc_node_metadata::time_create

3.13.1.5 quad_t cldc_node_metadata::time_modify

3.13.1.6 quad_t cldc_node_metadata::vers

The documentation for this struct was generated from the following file:

- include/cldc.h

3.14 cldc_ops Struct Reference

application-supplied facilities

```
#include <cldc.h>
```

Data Fields

- bool(∗ timer_ctl)(void ∗private, bool add, int(∗cb)(struct cldc_session ∗, void ∗), void ∗cb_private, time_t secs)
- int(∗ pkt_send)(void ∗private, const void ∗addr, size_t addrlen, const void ∗buf, size_t buflen)
- void(∗ event)(void ∗private, struct cldc_session ∗, struct cldc_fh ∗, uint32_t)

3.14.1 Detailed Description

application-supplied facilities

3.14.2 Field Documentation

3.14.2.1 void(∗ cldc_ops::event)(void ∗private, struct cldc_session ∗, struct cldc_fh ∗, uint32_t)

3.14.2.2 int(∗ cldc_ops::pkt_send)(void ∗private, const void ∗addr, size_t addrlen, const void ∗buf, size_t buflen)

3.14.2.3 bool(∗ cldc_ops::timer_ctl)(void ∗private, bool add, int(∗cb)(struct cldc_session ∗, void ∗), void ∗cb_private, time_t secs)

The documentation for this struct was generated from the following file:

- include/cldc.h

3.15 cldc_pkt_info Struct Reference

```
#include <cldc.h>
```

Data Fields

- int pkt_len
- int hdr_len
- int retries
- char user [CLD_MAX_USERNAME]
- char data [0]

3.15.1 Field Documentation

3.15.1.1 char cldc_pkt_info::data[0]

3.15.1.2 int cldc_pkt_info::hdr_len

3.15.1.3 int cldc_pkt_info::pkt_len

3.15.1.4 int cldc_pkt_info::retries

3.15.1.5 char cldc_pkt_info::user[CLD_MAX_USERNAME]

The documentation for this struct was generated from the following file:

- include/cldc.h

3.16 cldc_session Struct Reference

a single CLD client session

```
#include <cldc.h>
```

Data Fields

- uint8_t sid [CLD_SID_SZ]
- const struct cldc_ops ∗ ops
- struct hail_log log
- void ∗ private
- uint8_t addr [64]
- size_t addr_len
- GList ∗ cfh
- GList ∗ out_msg
- time_t msg_scan_time
- time_t expire_time
- bool expired
- uint64_t next_seqid_in
- uint64_t next_seqid_in_tr
- uint64_t next_seqid_out
- char user [CLD_MAX_USERNAME]
- char secret_key [CLD_MAX_SECRET_KEY]

- bool confirmed
- enum cld_msg_op msg_buf_op
- unsigned int msg_buf_len
- char msg_buf [CLD_MAX_MSG_SZ]
- char payload [CLD_MAX_PAYLOAD_SZ]
- char inode_name_temp [CLD_INODE_NAME_MAX]

3.16.1 Detailed Description

a single CLD client session

3.16.2 Field Documentation

3.16.2.1 uint8_t cldc_session::addr[64]

3.16.2.2 size_t cldc_session::addr_len

3.16.2.3 GList∗ cldc_session::cfh

3.16.2.4 bool cldc_session::confirmed

3.16.2.5 time_t cldc_session::expire_time

3.16.2.6 bool cldc_session::expired

3.16.2.7 char cldc_session::inode_name_temp[CLD_INODE_NAME_MAX]

3.16.2.8 struct hail_log cldc_session::log

3.16.2.9 char cldc_session::msg_buf[CLD_MAX_MSG_SZ]

3.16.2.10 unsigned int cldc_session::msg_buf_len

3.16.2.11 enum cld_msg_op cldc_session::msg_buf_op

3.16.2.12 time_t cldc_session::msg_scan_time

3.16.2.13 uint64_t cldc_session::next_seqid_in

3.16.2.14 uint64_t cldc_session::next_seqid_in_tr

3.16.2.15 uint64_t cldc_session::next_seqid_out

3.16.2.16 const struct cldc_ops∗ cldc_session::ops

3.16.2.17 GList∗ cldc_session::out_msg

3.16.2.18 char cldc_session::payload[CLD_MAX_PAYLOAD_SZ]

3.16.2.19 void∗ cldc_session::private

3.16.2.20 char cldc_session::secret_key[CLD_MAX_SECRET_KEY]

3.16.2.21 uint8_t cldc_session::sid[CLD_SID_SZ]

3.16.2.22 char cldc_session::user[CLD_MAX_USERNAME]

The documentation for this struct was generated from the following file:

- include/cldc.h

3.17 cldc_udp Struct Reference

A UDP implementation of the CLD client protocol.

```
#include <cldc.h>
```

Data Fields

- uint8_t addr [64]
- size_t addr_len
- int fd
- struct cldc_session ∗ sess
- int(∗ cb)(struct cldc_session ∗, void ∗)
- void ∗ cb_private

3.17.1 Detailed Description

A UDP implementation of the CLD client protocol.

3.17.2 Field Documentation

3.17.2.1 uint8_t cldc_udp::addr[64]

3.17.2.2 size_t cldc_udp::addr_len

3.17.2.3 int(∗ cldc_udp::cb)(struct **cldc_session** ∗, void ∗)

3.17.2.4 void∗ cldc_udp::cb_private

3.17.2.5 int cldc_udp::fd

3.17.2.6 struct **cldc_session**∗ cldc_udp::sess

The documentation for this struct was generated from the following file:

- include/cldc.h

3.18 hail_log Struct Reference

```
#include <hail_log.h>
```

Data Fields

- void(∗ func)(int prio, const char ∗fmt,...) ATTR_PRINTF(2
- void(∗) boo debug)
- bool verbose

3.18.1 Field Documentation

3.18.1.1 void(∗) boo hail_log::debug)

3.18.1.2 void(∗ hail_log::func)(int prio, const char ∗fmt,...) ATTR_PRINTF(2

3.18.1.3 bool hail_log::verbose

The documentation for this struct was generated from the following file:

- include/hail_log.h

3.19 hstor_blist Struct Reference

```
#include <hstor.h>
```

Data Fields

- char ∗ own_id
- char ∗ own_name
- GList ∗ list

3.19.1 Field Documentation

3.19.1.1 GList∗ hstor_blist::list

3.19.1.2 char∗ hstor_blist::own_id

3.19.1.3 char∗ hstor_blist::own_name

The documentation for this struct was generated from the following file:

- include/hstor.h

3.20 hstor_bucket Struct Reference

```
#include <hstor.h>
```

Data Fields

- char ∗ name
- char ∗ time_create

3.20.1 Field Documentation

3.20.1.1 char∗ hstor_bucket::name

3.20.1.2 char∗ hstor_bucket::time_create

The documentation for this struct was generated from the following file:

- include/hstor.h

3.21 hstor_client Struct Reference

```
#include <hstor.h>
```

Data Fields

- CURL ∗ curl
- char ∗ acc
- char ∗ host
- char ∗ user
- char ∗ key
- bool verbose
- bool subdomain

3.21.1 Field Documentation

3.21.1.1 char∗ hstor_client::acc

3.21.1.2 CURL∗ hstor_client::curl

3.21.1.3 char∗ hstor_client::host

3.21.1.4 char∗ hstor_client::key

3.21.1.5 bool hstor_client::subdomain

3.21.1.6 char∗ hstor_client::user

3.21.1.7 bool hstor_client::verbose

The documentation for this struct was generated from the following file:

- include/hstor.h

3.22 hstor_keylist Struct Reference

```
#include <hstor.h>
```

Data Fields

- char ∗ name
- char ∗ prefix
- char ∗ marker
- char ∗ delim
- unsigned int max_keys
- bool trunc
- GList ∗ contents
- GList ∗ common_pfx

3.22.1 Field Documentation

3.22.1.1 GList∗ hstor_keylist::common_pfx

3.22.1.2 GList∗ hstor_keylist::contents

3.22.1.3 char∗ hstor_keylist::delim

3.22.1.4 char∗ hstor_keylist::marker

3.22.1.5 unsigned int hstor_keylist::max_keys

3.22.1.6 char∗ hstor_keylist::name

3.22.1.7 char∗ hstor_keylist::prefix

3.22.1.8 bool hstor_keylist::trunc

The documentation for this struct was generated from the following file:

- include/hstor.h

3.23 hstor_object Struct Reference

```
#include <hstor.h>
```

Data Fields

- char ∗ key
- char ∗ time_mod
- char ∗ etag
- uint64_t size
- char ∗ storage
- char ∗ own_id
- char ∗ own_name

3.23.1 Field Documentation

3.23.1.1 char∗ hstor_object::etag

3.23.1.2 char∗ hstor_object::key

3.23.1.3 char∗ hstor_object::own_id

3.23.1.4 char∗ hstor_object::own_name

3.23.1.5 uint64_t hstor_object::size

3.23.1.6 char∗ hstor_object::storage

3.23.1.7 char∗ hstor_object::time_mod

The documentation for this struct was generated from the following file:

- include/hstor.h

3.24 http_hdr Struct Reference

```
#include <hstor.h>
```

Data Fields

- char ∗ key
- char ∗ val

3.24.1 Field Documentation

3.24.1.1 char∗ http_hdr::key

3.24.1.2 char∗ http_hdr::val

The documentation for this struct was generated from the following file:

- include/hstor.h

3.25 http_req Struct Reference

```
#include <hstor.h>
```

Data Fields

- char ∗ method
- struct http_uri uri
- int major
- int minor
- char ∗ orig_path
- unsigned int n_hdr
- struct http_hdr hdr [HREQ_MAX_HDR]

3.25.1 Field Documentation

3.25.1.1 struct http_hdr http_req::hdr[HREQ_MAX_HDR]

3.25.1.2 int http_req::major

3.25.1.3 char∗ http_req::method

3.25.1.4 int http_req::minor

3.25.1.5 unsigned int http_req::n_hdr

3.25.1.6 char∗ http_req::orig_path

3.25.1.7 struct **http_uri** http_req::uri

The documentation for this struct was generated from the following file:

- include/hstor.h

3.26 http_uri Struct Reference

```
#include <hstor.h>
```

Data Fields

- char ∗ scheme
- unsigned int scheme_len
- char ∗ userinfo
- unsigned int userinfo_len
- char ∗ hostname
- unsigned int hostname_len
- unsigned int port
- char ∗ path
- unsigned int path_len
- char ∗ query
- unsigned int query_len
- char ∗ fragment
- unsigned int fragment_len

3.26.1 Field Documentation

3.26.1.1 char∗ http_uri::fragment

3.26.1.2 unsigned int http_uri::fragment_len

3.26.1.3 char∗ http_uri::hostname

3.26.1.4 unsigned int http_uri::hostname_len

3.26.1.5 char∗ http_uri::path

3.26.1.6 unsigned int http_uri::path_len

3.26.1.7 unsigned int http_uri::port

3.26.1.8 char∗ http_uri::query

3.26.1.9 unsigned int http_uri::query_len

3.26.1.10 char∗ http_uri::scheme

3.26.1.11 unsigned int http_uri::scheme_len

3.26.1.12 char∗ http_uri::userinfo

3.26.1.13 unsigned int http_uri::userinfo_len

The documentation for this struct was generated from the following file:

- include/hstor.h

3.27 list_head Struct Reference

```
#include <elist.h>
```

Data Fields

- struct list_head ∗ next
- struct list_head ∗ prev

3.27.1 Field Documentation

3.27.1.1 struct **list_head**∗ list_head::next

3.27.1.2 struct **list_head** ∗ list_head::prev

The documentation for this struct was generated from the following file:

- include/elist.h

3.28 ncld_fh Struct Reference

```
#include <ncld.h>
```

Data Fields

- struct ncld_sess ∗ sess
- struct cldc_fh ∗ fh
- bool is_open
- int errc
- int nios
- unsigned int event_mask
- void(∗ event_func)(void ∗, unsigned int)
- void ∗ event_arg

3.28.1 Field Documentation

3.28.1.1 int ncld_fh::errc

3.28.1.2 void∗ ncld_fh::event_arg

3.28.1.3 void(∗ ncld_fh::event_func)(void ∗, unsigned int)

3.28.1.4 unsigned int ncld_fh::event_mask

3.28.1.5 struct **cldc_fh**∗ ncld_fh::fh

3.28.1.6 bool ncld_fh::is_open

3.28.1.7 int ncld_fh::nios

3.28.1.8 struct **ncld_sess**∗ ncld_fh::sess

The documentation for this struct was generated from the following file:

- include/ncld.h

3.29 ncld_read Struct Reference

```
#include <ncld.h>
```

Data Fields

- const void ∗ ptr
- long length
- struct cldc_node_metadata meta
- struct ncld_fh ∗ fh
- bool is_done
- int errc

3.29.1 Field Documentation

3.29.1.1 int ncld_read::errc

3.29.1.2 struct **ncld_fh**∗ ncld_read::fh

3.29.1.3 bool ncld_read::is_done

3.29.1.4 long ncld_read::length

3.29.1.5 struct **cldc_node_metadata** ncld_read::meta

3.29.1.6 const void∗ ncld_read::ptr

The documentation for this struct was generated from the following file:

- include/ncld.h

3.30 ncld_sess Struct Reference

```
#include <ncld.h>
```

Data Fields

- char ∗ host
- unsigned short port

- GMutex ∗ mutex
- GCond ∗ cond
- GThread ∗ thread
- bool is_up
- bool open_done
- int errc
- GList ∗ handles
- int to_thread [2]
- struct cldc_udp ∗ udp
- struct cld_timer udp_timer
- struct cld_timer_list tlist
- void(∗ event)(void ∗, unsigned int)
- void ∗ event_arg

3.30.1 Field Documentation

3.30.1.1 GCond∗ ncld_sess::cond

3.30.1.2 int ncld_sess::errc

3.30.1.3 void(∗ ncld_sess::event)(void ∗, unsigned int)

3.30.1.4 void∗ ncld_sess::event_arg

3.30.1.5 GList∗ ncld_sess::handles

3.30.1.6 char∗ ncld_sess::host

3.30.1.7 bool ncld_sess::is_up

3.30.1.8 GMutex∗ ncld_sess::mutex

3.30.1.9 bool ncld_sess::open_done

3.30.1.10 unsigned short ncld_sess::port

3.30.1.11 GThread∗ ncld_sess::thread

3.30.1.12 struct **cld_timer_list** ncld_sess::tlist

3.30.1.13 int ncld_sess::to_thread[2]

3.30.1.14 struct **cldc_udp**∗ ncld_sess::udp

3.30.1.15 struct **cld_timer** ncld_sess::udp_timer

The documentation for this struct was generated from the following file:

- include/ncld.h

3.31 objcache Struct Reference

```
#include <objcache.h>
```

Data Fields

- GMutex ∗ lock
- GHashTable ∗ table

3.31.1 Field Documentation

3.31.1.1 GMutex∗ objcache::lock

3.31.1.2 GHashTable∗ objcache::table

The documentation for this struct was generated from the following file:

- include/objcache.h

3.32 objcache_entry Struct Reference

```
#include <objcache.h>
```

Data Fields

- unsigned int hash
- unsigned int flags
- int ref

3.32.1 Field Documentation

3.32.1.1 unsigned int objcache_entry::flags

3.32.1.2 unsigned int objcache_entry::hash

3.32.1.3 int objcache_entry::ref

The documentation for this struct was generated from the following file:

- include/objcache.h

3.33 st_client Struct Reference

```
#include <chunkc.h>
```

Data Fields

- char ∗ host
- char ∗ user
- char ∗ key
- bool verbose
- int fd
- SSL_CTX ∗ ssl_ctx
- SSL ∗ ssl
- char req_buf [sizeof(struct chunksrv_req)+CHD_KEY_SZ]

3.33.1 Field Documentation

3.33.1.1 int st_client::fd

3.33.1.2 char* st_client::host

3.33.1.3 char* st_client::key

3.33.1.4 char st_client::req_buf[sizeof(struct chunksrv_req)+CHD_KEY_SZ]

3.33.1.5 SSL* st_client::ssl

3.33.1.6 SSL_CTX* st_client::ssl_ctx

3.33.1.7 char* st_client::user

3.33.1.8 bool st_client::verbose

The documentation for this struct was generated from the following file:

- include/chunkc.h

3.34 st_keylist Struct Reference

```
#include <chunkc.h>
```

Data Fields

- char * name
- GList * contents

3.34.1 Field Documentation

3.34.1.1 GList* st_keylist::contents

3.34.1.2 char* st_keylist::name

The documentation for this struct was generated from the following file:

- include/chunkc.h

3.35 st_object Struct Reference

```
#include <chunkc.h>
```

Data Fields

- char * name
- char * time_mod
- char * etag
- uint64_t size
- char * owner

3.35.1 Field Documentation

3.35.1.1 char∗ st_object::etag

3.35.1.2 char∗ st_object::name

3.35.1.3 char∗ st_object::owner

3.35.1.4 uint64_t st_object::size

3.35.1.5 char∗ st_object::time_mod

The documentation for this struct was generated from the following file:

- include/chunkc.h

Chapter 4

File Documentation

4.1 include/chunk-private.h File Reference

```
#include <stdint.h>
#include <glib.h>
```

Macros

- #define MDB_TPATH_FMT "%s/%X"
- #define BAD_TPATH_FMT "%s/bad"
- #define PREFIX_LEN 3

4.1.1 Macro Definition Documentation

4.1.1.1 #define BAD_TPATH_FMT "%s/bad"

4.1.1.2 #define MDB_TPATH_FMT "%s/%X"

4.1.1.3 #define PREFIX_LEN 3

4.2 include/chunk_msg.h File Reference

```
#include <stdint.h>
```

Data Structures

- struct chunksrv_req
- struct chunksrv_resp
- struct chunksrv_resp_get
- struct chunk_check_status
- struct chunksrv_resp_chkstat

Macros

- #define CHUNKD_MAGIC "CHUNKDv1"

Enumerations

- enum {
  CHD_MAGIC_SZ = 8, CHD_USER_SZ = 64, CHD_KEY_SZ = 1024, CHD_CSUM_SZ = 20,
  CHD_SIG_SZ = 64 }
- enum chunksrv_ops {
  CHO_NOP = 0, CHO_GET = 1, CHO_GET_META = 2, CHO_PUT = 3,
  CHO_DEL = 4, CHO_LIST = 5, CHO_LOGIN = 6, CHO_TABLE_OPEN = 7,
  CHO_CHECK_START = 8, CHO_CHECK_STATUS = 9, CHO_START_TLS = 10, CHO_CP = 11 }
- enum chunk_errcode {
  che_Success = 0, che_AccessDenied = 1, che_InternalError = 2, che_InvalidArgument = 3,
  che_InvalidURI = 4, che_NoSuchKey = 5, che_SignatureDoesNotMatch = 6, che_InvalidKey = 7,
  che_InvalidTable = 8, che_Busy = 9, che_KeyExists = 10 }
- enum chunk_flags { CHF_SYNC = (1 $<<$ 0), CHF_TBL_CREAT = (1 $<<$ 1), CHF_TBL_EXCL = (1 $<<$ 2) }
- enum chunk_check_state { chk_Off, chk_Idle, chk_Active }

4.2.1 Macro Definition Documentation

4.2.1.1 #define CHUNKD_MAGIC "CHUNKDv1"

4.2.2 Enumeration Type Documentation

4.2.2.1 anonymous enum

Enumerator

CHD_MAGIC_SZ

CHD_USER_SZ

CHD_KEY_SZ

CHD_CSUM_SZ

CHD_SIG_SZ

4.2.2.2 enum chunk_check_state

Enumerator

chk_Off

chk_Idle

chk_Active

4.2.2.3 enum chunk_errcode

Enumerator

che_Success

che_AccessDenied

che_InternalError

che_InvalidArgument

che_InvalidURI

che_NoSuchKey

che_SignatureDoesNotMatch

che_InvalidKey

che_InvalidTable

che_Busy

che_KeyExists

4.2.2.4 enum chunk_flags

Enumerator

CHF_SYNC

CHF_TBL_CREAT

CHF_TBL_EXCL

4.2.2.5 enum chunksrv_ops

Enumerator

CHO_NOP

CHO_GET

CHO_GET_META

CHO_PUT

CHO_DEL

CHO_LIST

CHO_LOGIN

CHO_TABLE_OPEN

CHO_CHECK_START

CHO_CHECK_STATUS

CHO_START_TLS

CHO_CP

4.3 include/chunkc.h File Reference

```
#include <sys/types.h>
#include <openssl/ssl.h>
#include <stdbool.h>
#include <stdint.h>
#include <string.h>
#include <glib.h>
#include <chunk_msg.h>
```

Data Structures

- struct st_object
- struct st_keylist
- struct st_client

Functions

- void stc_free (struct st_client ∗stc)
- void stc_free_keylist (struct st_keylist ∗keylist)
- void stc_free_object (struct st_object ∗obj)
- void stc_init (void)
- struct st_client ∗ stc_new (const char ∗service_host, int port, const char ∗user, const char ∗secret_key, bool encrypt)
- bool stc_table_open (struct st_client ∗stc, const void ∗key, size_t key_len, uint32_t flags)
- bool stc_get (struct st_client ∗stc, const void ∗key, size_t key_len, size_t(∗write_cb)(void ∗, size_t, size_t, void ∗), void ∗user_data)
- void ∗ stc_get_inline (struct st_client ∗stc, const void ∗key, size_t key_len, size_t ∗len)
- bool stc_get_start (struct st_client ∗stc, const void ∗key, size_t key_len, int ∗pfd, uint64_t ∗len)
- size_t stc_get_recv (struct st_client ∗stc, void ∗data, size_t len)
- bool stc_put (struct st_client ∗stc, const void ∗key, size_t key_len, size_t(∗read_cb)(void ∗, size_t, size_t, void ∗), uint64_t len, void ∗user_data, uint32_t flags)
- bool stc_put_start (struct st_client ∗stc, const void ∗key, size_t key_len, uint64_t cont_len, int ∗pfd, uint32_t flags)
- size_t stc_put_send (struct st_client ∗stc, void ∗data, size_t len)
- bool stc_put_sync (struct st_client ∗stc)
- bool stc_put_inline (struct st_client ∗stc, const void ∗key, size_t key_len, void ∗data, uint64_t len, uint32_t flags)
- bool stc_cp (struct st_client ∗stc, const void ∗dest_key, size_t dest_key_len, const void ∗src_key, size_t src↵ _key_len)
- bool stc_del (struct st_client ∗stc, const void ∗key, size_t key_len)
- bool stc_ping (struct st_client ∗stc)
- bool stc_check_start (struct st_client ∗stc)
- bool stc_check_status (struct st_client ∗stc, struct chunk_check_status ∗out)
- struct st_keylist ∗ stc_keys (struct st_client ∗stc)
- int stc_readport (const char ∗fname)

4.3.1 Function Documentation

4.3.1.1 bool stc_check_start (struct **st_client** ∗ *stc*)

4.3.1.2 bool stc_check_status (struct **st_client** ∗ *stc,* struct **chunk_check_status** ∗ *out*)

4.3.1.3 bool stc_cp (struct **st_client** ∗ *stc,* const void ∗ *dest_key,* size_t *dest_key_len,* const void ∗ *src_key,* size_t *src_key_len*)

4.3.1.4 bool stc_del (struct **st_client** ∗ *stc,* const void ∗ *key,* size_t *key_len*)

4.3.1.5 void stc_free (struct **st_client** ∗ *stc*)

4.3.1.6 void stc_free_keylist (struct **st_keylist** ∗ *keylist*)

4.3.1.7 void stc_free_object (struct **st_object** ∗ *obj*)

4.3.1.8 bool stc_get (struct **st_client** ∗ *stc,* const void ∗ *key,* size_t *key_len,* size_t(∗)(void ∗, size_t, size_t, void ∗) *write_cb,* void ∗ *user_data*)

4.3.1.9 void∗ stc_get_inline (struct **st_client** ∗ *stc,* const void ∗ *key,* size_t *key_len,* size_t ∗ *len*)

4.3.1.10 size_t stc_get_recv (struct **st_client** ∗ *stc,* void ∗ *data,* size_t *len*)

4.3.1.11 bool stc_get_start (struct **st_client** ∗ *stc,* const void ∗ *key,* size_t *key_len,* int ∗ *pfd,* uint64_t ∗ *len*)

4.3.1.12 void stc_init (void)

4.3.1.13 struct **st_keylist**∗ stc_keys (struct **st_client** ∗ *stc*)

4.3.1.14 struct **st_client**∗ stc_new (const char ∗ *service_host,* int *port,* const char ∗ *user,* const char ∗ *secret_key,* bool *encrypt*)

4.3.1.15 bool stc_ping (struct **st_client** ∗ *stc*)

4.3.1.16 bool stc_put (struct **st_client** ∗ *stc,* const void ∗ *key,* size_t *key_len,* size_t(∗)(void ∗, size_t, size_t, void ∗) *read_cb,* uint64_t *len,* void ∗ *user_data,* uint32_t *flags*)

4.3.1.17 bool stc_put_inline (struct **st_client** ∗ *stc,* const void ∗ *key,* size_t *key_len,* void ∗ *data,* uint64_t *len,* uint32_t *flags*)

4.3.1.18 size_t stc_put_send (struct **st_client** ∗ *stc,* void ∗ *data,* size_t *len*)

4.3.1.19 bool stc_put_start (struct **st_client** ∗ *stc,* const void ∗ *key,* size_t *key_len,* uint64_t *cont_len,* int ∗ *pfd,* uint32_t *flags*)

4.3.1.20 bool stc_put_sync (struct **st_client** ∗ *stc*)

4.3.1.21 int stc_readport (const char ∗ *fname*)

4.3.1.22 bool stc_table_open (struct **st_client** ∗ *stc,* const void ∗ *key,* size_t *key_len,* uint32_t *flags*)

4.4 include/chunksrv.h File Reference

```
#include <chunk_msg.h>
```

Functions

- size_t req_len (const struct chunksrv_req ∗req)
- void chreq_sign (struct chunksrv_req ∗req, const char ∗key, char ∗b64hmac_out)

4.4.1 Function Documentation

4.4.1.1 void chreq_sign (struct **chunksrv_req** ∗ *req,* const char ∗ *key,* char ∗ *b64hmac_out*)

4.4.1.2 size_t req_len (const struct **chunksrv_req** ∗ *req*)

4.5 include/cld-private.h File Reference

```
#include <stdint.h>
#include <glib.h>
```

4.6 include/cld_common.h File Reference

```
#include <stdint.h>
#include <stdbool.h>
#include <string.h>
#include <time.h>
#include <glib.h>
#include <openssl/sha.h>
#include <cld_msg_rpc.h>
```

Data Structures

- struct cld_timer
- struct cld_timer_list

Macros

- #define CLD_ALIGN8(n) ((8 - ((n) & 7)) & 7)
- #define SIDFMT "%016llX"
- #define SIDARG(sid) cld_sid2llu(sid)
- #define CLD_PKT_FTR_LEN sizeof(struct cld_pkt_ftr)

    Length of the packet footer.
- #define PKT_HDR_TO_STR_SCRATCH_LEN 128

Functions

- void cld_timer_add (struct cld_timer_list ∗tlist, struct cld_timer ∗timer, time_t expires)
- void cld_timer_del (struct cld_timer_list ∗tlist, struct cld_timer ∗timer)
- time_t cld_timers_run (struct cld_timer_list ∗tlist)
- unsigned long long cld_sid2llu (const uint8_t ∗sid)
- void cld_rand64 (void ∗p)
- const char ∗ cld_errstr (enum cle_err_codes ecode)
- int cld_readport (const char ∗fname)
- int cld_authcheck (struct hail_log ∗log, const char ∗key, const void ∗buf, size_t buf_len, const void ∗sha)
- int cld_authsign (struct hail_log ∗log, const char ∗key, const void ∗buf, size_t buf_len, void ∗sha)
- const char ∗ cld_opstr (enum cld_msg_op)
- const char ∗ cld_pkt_hdr_to_str (char ∗scratch, const char ∗pkt_hdr, size_t pkt_len)
- void __cld_dump_buf (const void ∗buf, size_t len)
- struct __attribute__ ((packed)) cld_pkt_ftr

    Footer that appears at the end of each packet.

4.6.1 Macro Definition Documentation

4.6.1.1 #define CLD_ALIGN8(*n*) ((8 - ((n) & 7)) & 7)

4.6.1.2 #define CLD_PKT_FTR_LEN sizeof(struct cld_pkt_ftr)

Length of the packet footer.

This size is fixed

4.6.1.3 #define PKT_HDR_TO_STR_SCRATCH_LEN 128

4.6.1.4 #define SIDARG(*sid*) cld_sid2llu(sid)

4.6.1.5 #define SIDFMT "%016llX"

4.6.2 Function Documentation

4.6.2.1 struct __attribute__ ((packed))

Footer that appears at the end of each packet.

< packet sequence ID

< packet signature

4.6.2.2 void __cld_dump_buf (const void ∗ *buf,* size_t *len*)

4.6.2.3 int cld_authcheck (struct hail_log ∗ *log,* const char ∗ *key,* const void ∗ *buf,* size_t *buf_len,* const void ∗ *sha*)

4.6.2.4 int cld_authsign (struct hail_log ∗ *log,* const char ∗ *key,* const void ∗ *buf,* size_t *buf_len,* void ∗ *sha*)

4.6.2.5 const char∗ cld_errstr (enum cle_err_codes *ecode*)

4.6.2.6 const char∗ cld_opstr (enum *cld_msg_op*)

4.6.2.7 const char∗ cld_pkt_hdr_to_str (char ∗ *scratch,* const char ∗ *pkt_hdr,* size_t *pkt_len*)

4.6.2.8 void cld_rand64 (void ∗ *p*)

4.6.2.9 int cld_readport (const char ∗ *fname*)

4.6.2.10 unsigned long long cld_sid2llu (const uint8_t ∗ *sid*)

4.6.2.11 void cld_timer_add (struct cld_timer_list ∗ *tlist,* struct cld_timer ∗ *timer,* time_t *expires*)

4.6.2.12 void cld_timer_del (struct cld_timer_list ∗ *tlist,* struct cld_timer ∗ *timer*)

4.6.2.13 time_t cld_timers_run (struct cld_timer_list ∗ *tlist*)

4.7 include/cldc.h File Reference

```
#include <sys/types.h>
#include <stdbool.h>
#include <glib.h>
#include <cld_msg_rpc.h>
#include <cld_common.h>
#include <hail_log.h>
```

Data Structures

- struct cldc_call_opts

  per-operation application options
- struct cldc_node_metadata

- struct cldc_pkt_info
- struct cldc_msg

  an outgoing message, from client to server
- struct cldc_fh

  an open file handle associated with a session
- struct cldc_ops

  application-supplied facilities
- struct cldc_session

  a single CLD client session
- struct cldc_host

  Information for a single CLD server host.
- struct cldc_udp

  A UDP implementation of the CLD client protocol.
- struct cld_dirent_cur

Functions

- int cldc_receive_pkt (struct cldc_session ∗sess, const void ∗net_addr, size_t net_addrlen, const void ∗buf, size_t buflen)

  Packet received from remote host.
- void cldc_init (void)
- int cldc_new_sess (const struct cldc_ops ∗ops, const struct cldc_call_opts ∗copts, const void ∗addr, size_t addr_len, const char ∗user, const char ∗secret_key, void ∗private, struct cldc_session ∗∗sess_out)
- void cldc_kill_sess (struct cldc_session ∗sess)
- int cldc_end_sess (struct cldc_session ∗sess, const struct cldc_call_opts ∗copts)
- int cldc_nop (struct cldc_session ∗sess, const struct cldc_call_opts ∗copts)
- int cldc_del (struct cldc_session ∗sess, const struct cldc_call_opts ∗copts, const char ∗pathname)
- int cldc_open (struct cldc_session ∗sess, const struct cldc_call_opts ∗copts, const char ∗pathname, uint32_t open_mode, uint32_t events, struct cldc_fh ∗∗fh_out)
- int cldc_close (struct cldc_fh ∗fh, const struct cldc_call_opts ∗copts)
- int cldc_unlock (struct cldc_fh ∗fh, const struct cldc_call_opts ∗copts)
- int cldc_lock (struct cldc_fh ∗fh, const struct cldc_call_opts ∗copts, uint32_t lock_flags, bool wait_for_lock)
- int cldc_put (struct cldc_fh ∗fh, const struct cldc_call_opts ∗copts, const void ∗data, size_t data_len)
- int cldc_get (struct cldc_fh ∗fh, const struct cldc_call_opts ∗copts, bool metadata_only)
- int cldc_dirent_count (const void ∗data, size_t data_len)
- int cldc_dirent_first (struct cld_dirent_cur ∗dc)
- int cldc_dirent_next (struct cld_dirent_cur ∗dc)
- void cldc_dirent_cur_init (struct cld_dirent_cur ∗dc, const void ∗buf, size_t buflen)
- void cldc_dirent_cur_fini (struct cld_dirent_cur ∗dc)
- char ∗ cldc_dirent_name (struct cld_dirent_cur ∗dc)
- void cldc_copts_get_data (const struct cldc_call_opts ∗copts, char ∗∗data, size_t ∗data_len)
- void cldc_copts_get_metadata (const struct cldc_call_opts ∗copts, struct cldc_node_metadata ∗md)
- void cldc_udp_free (struct cldc_udp ∗udp)
- int cldc_udp_new (const char ∗hostname, int port, struct cldc_udp ∗∗udp_out)
- int cldc_udp_receive_pkt (struct cldc_udp ∗udp)
- int cldc_udp_pkt_send (void ∗private, const void ∗addr, size_t addrlen, const void ∗buf, size_t buflen)
- int cldc_getaddr (GList ∗∗host_list, const char ∗thishost, struct hail_log ∗log)
- int cldc_saveaddr (struct cldc_host ∗hp, unsigned int priority, unsigned int weight, unsigned int port, unsigned int nlen, const char ∗name, struct hail_log ∗log)

4.7.1 Function Documentation

4.7.1.1 int cldc_close (struct cldc_fh ∗ *fh,* const struct cldc_call_opts ∗ *copts*)

4.7.1.2 void cldc_copts_get_data (const struct cldc_call_opts ∗ *copts,* char ∗∗ *data,* size_t ∗ *data_len*)

4.7.1.3 void cldc_copts_get_metadata (const struct cldc_call_opts ∗ *copts,* struct cldc_node_metadata ∗ *md*)

4.7.1.4 int cldc_del (struct cldc_session ∗ *sess,* const struct cldc_call_opts ∗ *copts,* const char ∗ *pathname*)

4.7.1.5 int cldc_dirent_count (const void ∗ *data,* size_t *data_len*)

4.7.1.6 void cldc_dirent_cur_fini (struct cld_dirent_cur ∗ *dc*)

4.7.1.7 void cldc_dirent_cur_init (struct cld_dirent_cur ∗ *dc,* const void ∗ *buf,* size_t *buflen*)

4.7.1.8 int cldc_dirent_first (struct cld_dirent_cur ∗ *dc*)

4.7.1.9 char∗ cldc_dirent_name (struct cld_dirent_cur ∗ *dc*)

4.7.1.10 int cldc_dirent_next (struct cld_dirent_cur ∗ *dc*)

4.7.1.11 int cldc_end_sess (struct cldc_session ∗ *sess,* const struct cldc_call_opts ∗ *copts*)

4.7.1.12 int cldc_get (struct cldc_fh ∗ *fh,* const struct cldc_call_opts ∗ *copts,* bool *metadata_only*)

4.7.1.13 int cldc_getaddr (GList ∗∗ *host_list,* const char ∗ *thishost,* struct hail_log ∗ *log*)

4.7.1.14 void cldc_init (void)

4.7.1.15 void cldc_kill_sess (struct cldc_session ∗ *sess*)

4.7.1.16 int cldc_lock (struct cldc_fh ∗ *fh,* const struct cldc_call_opts ∗ *copts,* uint32_t *lock_flags,* bool *wait_for_lock*)

4.7.1.17 int cldc_new_sess (const struct cldc_ops ∗ *ops,* const struct cldc_call_opts ∗ *copts,* const void ∗ *addr,* size_t *addr_len,* const char ∗ *user,* const char ∗ *secret_key,* void ∗ *private,* struct cldc_session ∗∗ *sess_out*)

4.7.1.18 int cldc_nop (struct cldc_session ∗ *sess,* const struct cldc_call_opts ∗ *copts*)

4.7.1.19 int cldc_open (struct cldc_session ∗ *sess,* const struct cldc_call_opts ∗ *copts,* const char ∗ *pathname,* uint32_t *open_mode,* uint32_t *events,* struct cldc_fh ∗∗ *fh_out*)

4.7.1.20 int cldc_put (struct cldc_fh ∗ *fh,* const struct cldc_call_opts ∗ *copts,* const void ∗ *data,* size_t *data_len*)

4.7.1.21 int cldc_receive_pkt (struct cldc_session ∗ *sess,* const void ∗ *net_addr,* size_t *net_addrlen,* const void ∗ *buf,* size_t *buflen*)

Packet received from remote host.

Called by app when a packet is received from a remote host over the network.

Parameters

sess	Session associated with received packet

net_addr	Opaque network address
net_addrlen	Size of opaque network address
buf	Pointer to data buffer containing packet
buflen	Length of received packet

Returns

Zero for success, non-zero on error

4.7.1.22 int cldc_saveaddr (struct **cldc_host** ∗ *hp*, unsigned int *priority*, unsigned int *weight*, unsigned int *port*, unsigned int *nlen*, const char ∗ *name*, struct **hail_log** ∗ *log*)

4.7.1.23 void cldc_udp_free (struct **cldc_udp** ∗ *udp*)

4.7.1.24 int cldc_udp_new (const char ∗ *hostname*, int *port*, struct **cldc_udp** ∗∗ *udp_out*)

4.7.1.25 int cldc_udp_pkt_send (void ∗ *private*, const void ∗ *addr*, size_t *addrlen*, const void ∗ *buf*, size_t *buflen*)

4.7.1.26 int cldc_udp_receive_pkt (struct **cldc_udp** ∗ *udp*)

4.7.1.27 int cldc_unlock (struct **cldc_fh** ∗ *fh*, const struct **cldc_call_opts** ∗ *copts*)

4.8 include/elist.h File Reference

Data Structures

- struct list_head

Macros

- #define LIST_HEAD_INIT(name) { &(name), &(name) }
- #define LIST_HEAD(name) struct list_head name = LIST_HEAD_INIT(name)
- #define INIT_LIST_HEAD(ptr)
- #define list_entry(ptr, type, member) ((type ∗)((char ∗)(ptr)-(unsigned long)(&((type ∗)0)->member)))

  list_entry - get the struct for this entry : the &struct list_head pointer.
- #define list_for_each(pos, head)

  list_for_each - iterate over a list : the &struct list_head to use as a loop counter.
- #define list_for_each_prev(pos, head)

  list_for_each_prev - iterate over a list backwards : the &struct list_head to use as a loop counter.
- #define list_for_each_safe(pos, n, head)

  list_for_each_safe - iterate over a list safe against removal of list entry : the &struct list_head to use as a loop counter.
- #define list_for_each_entry(pos, head, member)

  list_for_each_entry - iterate over list of given type : the type ∗ to use as a loop counter.
- #define list_for_each_entry_safe(pos, n, head, member)

  list_for_each_entry_safe - iterate over list of given type safe against removal of list entry : the type ∗ to use as a loop counter.
- #define list_for_each_entry_continue(pos, head, member)

  list_for_each_entry_continue - iterate over list of given type continuing after existing point : the type ∗ to use as a loop counter.

4.8.1 Macro Definition Documentation

4.8.1.1 #define INIT_LIST_HEAD(*ptr*)

Value:

```
do { \
        (ptr)->next = (ptr); (ptr)->prev = (ptr); \
} while (0)
```

4.8.1.2 #define list_entry(*ptr,* *type,* *member*) ((type ∗)((char ∗)(ptr)-(unsigned long)(&((type ∗)0)->member)))

list_entry - get the struct for this entry : the &struct list_head pointer.

: the type of the struct this is embedded in. : the name of the list_struct within the struct.

4.8.1.3 #define list_for_each(*pos,* *head*)

Value:

```
for (pos = (head)->next; pos != (head); \
                pos = pos->next)
```

list_for_each - iterate over a list : the &struct list_head to use as a loop counter.

: the head for your list.

4.8.1.4 #define list_for_each_entry(*pos,* *head,* *member*)

Value:

```
for (pos = list_entry((head)->next, typeof(*pos), member);    \
             &pos->member != (head);                          \
             pos = list_entry(pos->member.next, typeof(*pos), member))
```

list_for_each_entry - iterate over list of given type : the type ∗ to use as a loop counter.

: the head for your list. : the name of the list_struct within the struct.

4.8.1.5 #define list_for_each_entry_continue(*pos,* *head,* *member*)

Value:

```
for (pos = list_entry(pos->member.next, typeof(*pos), member),        \
                     prefetch(pos->member.next);                        \
             &pos->member != (head);                                    \
             pos = list_entry(pos->member.next, typeof(*pos), member),        \
                     prefetch(pos->member.next))
```

list_for_each_entry_continue - iterate over list of given type continuing after existing point : the type ∗ to use as a loop counter.

: the head for your list. : the name of the list_struct within the struct.

4.8.1.6 #define list_for_each_entry_safe(*pos,* *n,* *head,* *member*)

Value:

```
for (pos = list_entry((head)->next, typeof(*pos), member),     \
                n = list_entry(pos->member.next, typeof(*pos), member);        \
              &pos->member != (head);                                        \
              pos = n, n = list_entry(n->member.next, typeof(*n), member))
```

list_for_each_entry_safe - iterate over list of given type safe against removal of list entry : the type ∗ to use as a loop counter.

: another type ∗ to use as temporary storage : the head for your list. : the name of the list_struct within the struct.

4.8.1.7 #define list_for_each_prev(*pos,* *head*)

Value:

```
for (pos = (head)->prev; pos != (head); \
                pos = pos->prev)
```

list_for_each_prev - iterate over a list backwards : the &struct list_head to use as a loop counter.

: the head for your list.

4.8.1.8 #define list_for_each_safe(*pos,* *n,* *head*)

Value:

```
for (pos = (head)->next, n = pos->next; pos != (head); \
                pos = n, n = pos->next)
```

list_for_each_safe - iterate over a list safe against removal of list entry : the &struct list_head to use as a loop counter.

: another &struct list_head to use as temporary storage : the head for your list.

4.8.1.9 #define LIST_HEAD(*name*) struct **list_head** name = **LIST_HEAD_INIT**(name)

4.8.1.10 #define LIST_HEAD_INIT(*name*) { &(name), &(name) }

4.9 include/hail_log.h File Reference

```
#include <stdbool.h>
```

Data Structures

- struct hail_log

Macros

- #define ATTR_PRINTF(x, y)
- #define HAIL_VERBOSE(log,...)

  Print out a CLD session debug message if enabled.
- #define HAIL_DEBUG(log,...)

  Print out an application debug message if enabled.

- #define HAIL_INFO(log,...) (log)->func(LOG_INFO, __VA_ARGS__)

  Print out an informational log message.
- #define HAIL_WARN(log,...) (log)->func(LOG_WARNING, __VA_ARGS__)

  Print out a warning message.
- #define HAIL_ERR(log,...) (log)->func(LOG_ERR, __VA_ARGS__)

  Print out an error message.
- #define HAIL_CRIT(log,...) (log)->func(LOG_CRIT, __VA_ARGS__)

  Print out a critical warning message.

4.9.1 Macro Definition Documentation

4.9.1.1 #define ATTR_PRINTF(*x*, *y*)

4.9.1.2 #define HAIL_CRIT(*log*, *...*) (log)->func(LOG_CRIT, __VA_ARGS__)

Print out a critical warning message.

4.9.1.3 #define HAIL_DEBUG(*log*, *...*)

Value:

```
if ((log)->debug) { \
                (log)->func(LOG_DEBUG, __VA_ARGS__); \
        }
```

Print out an application debug message if enabled.

4.9.1.4 #define HAIL_ERR(*log*, *...*) (log)->func(LOG_ERR, __VA_ARGS__)

Print out an error message.

4.9.1.5 #define HAIL_INFO(*log*, *...*) (log)->func(LOG_INFO, __VA_ARGS__)

Print out an informational log message.

4.9.1.6 #define HAIL_VERBOSE(*log*, *...*)

Value:

```
if ((log)->verbose) { \
                (log)->func(LOG_DEBUG, __VA_ARGS__); \
        }
```

Print out a CLD session debug message if enabled.

4.9.1.7 #define HAIL_WARN(*log*, *...*) (log)->func(LOG_WARNING, __VA_ARGS__)

Print out a warning message.

4.10 include/hail_private.h File Reference

```
#include "hail-config.h"
#include <rpc/xdr.h>
```

4.11 include/hstor.h File Reference

```
#include <stdbool.h>
#include <stdint.h>
#include <curl/curl.h>
#include <glib.h>
```

Data Structures

- struct hstor_client
- struct hstor_bucket
- struct hstor_blist
- struct hstor_object
- struct hstor_keylist
- struct http_uri
- struct http_hdr
- struct http_req

Macros

- #define ARRAY_SIZE(arr) (sizeof(arr) / sizeof((arr)[0]))
- #define PATH_ESCAPE_MASK 0x02
- #define QUERY_ESCAPE_MASK 0x04

Enumerations

- enum hstor_calling_format { HFMT_ORDINARY, HFMT_SUBDOMAIN }
- enum { HREQ_MAX_HDR = 128 }
- enum ReqQ {
  URIQ_ACL, URIQ_LOCATION, URIQ_LOGGING, URIQ_TORRENT,
  URIQNUM }
- enum ReqACLC {
  ACLC_PRIV, ACLC_PUB_R, ACLC_PUB_RW, ACLC_AUTH_R,
  ACLCNUM }

Functions

- char ∗ hutil_time2str (char ∗buf, int len, time_t time)
- time_t hutil_str2time (const char ∗timestr)
- int hreq_hdr_push (struct http_req ∗req, char ∗key, char ∗val)
- char ∗ hreq_hdr (struct http_req ∗req, const char ∗key)
- void hreq_sign (struct http_req ∗req, const char ∗bucket, const char ∗key, char ∗b64hmac_out)
- GHashTable ∗ hreq_query (struct http_req ∗req)
- int hreq_is_query (struct http_req ∗req)
- void hreq_free (struct http_req ∗req)
- int hreq_acl_canned (struct http_req ∗req)
- struct http_uri ∗ huri_parse (struct http_uri ∗uri_dest, char ∗uri_src_text)
- int huri_field_unescape (char ∗s, int s_len)
- char ∗ huri_field_escape (const char ∗signed_str, unsigned char mask)
- void hstor_free (struct hstor_client ∗hstor)
- void hstor_free_blist (struct hstor_blist ∗blist)
- void hstor_free_bucket (struct hstor_bucket ∗buck)

- void hstor_free_object (struct hstor_object ∗obj)
- void hstor_free_keylist (struct hstor_keylist ∗keylist)
- struct hstor_client ∗ hstor_new (const char ∗service_acc, const char ∗service_host, const char ∗user, const char ∗secret_key)
- bool hstor_set_format (struct hstor_client ∗hstor, enum hstor_calling_format f)
- bool hstor_add_bucket (struct hstor_client ∗hstor, const char ∗name)
- bool hstor_del_bucket (struct hstor_client ∗hstor, const char ∗name)
- struct hstor_blist ∗ hstor_list_buckets (struct hstor_client ∗hstor)
- bool hstor_get (struct hstor_client ∗hstor, const char ∗bucket, const char ∗key, size_t(∗write_cb)(void ∗, size↵ _t, size_t, void ∗), void ∗user_data, bool want_headers)
- void ∗ hstor_get_inline (struct hstor_client ∗hstor, const char ∗bucket, const char ∗key, bool want_headers, size_t ∗len)
- bool hstor_put (struct hstor_client ∗hstor, const char ∗bucket, const char ∗key, size_t(∗read_cb)(void ∗, size↵ _t, size_t, void ∗), uint64_t len, void ∗user_data, char ∗∗user_hdrs)
- bool hstor_put_inline (struct hstor_client ∗hstor, const char ∗bucket, const char ∗key, void ∗data, uint64_t len, char ∗∗user_hdrs)
- bool hstor_del (struct hstor_client ∗hstor, const char ∗bucket, const char ∗key)
- struct hstor_keylist ∗ hstor_keys (struct hstor_client ∗hstor, const char ∗bucket, const char ∗prefix, const char ∗marker, const char ∗delim, unsigned int max_keys)

4.11.1 Macro Definition Documentation

4.11.1.1 #define ARRAY_SIZE(*arr*) (sizeof(arr) / sizeof((arr)[0]))

4.11.1.2 #define PATH_ESCAPE_MASK 0x02

4.11.1.3 #define QUERY_ESCAPE_MASK 0x04

4.11.2 Enumeration Type Documentation

4.11.2.1 anonymous enum

Enumerator

HREQ_MAX_HDR

4.11.2.2 enum hstor_calling_format

Enumerator

HFMT_ORDINARY

HFMT_SUBDOMAIN

4.11.2.3 enum ReqACLC

Enumerator

ACLC_PRIV

ACLC_PUB_R

ACLC_PUB_RW

ACLC_AUTH_R

ACLCNUM

4.11.2.4 enum **ReqQ**

Enumerator

URIQ_ACL

URIQ_LOCATION

URIQ_LOGGING

URIQ_TORRENT

URIQNUM

4.11.3 Function Documentation

4.11.3.1 int hreq_acl_canned (struct **http_req** ∗ *req*)

4.11.3.2 void hreq_free (struct **http_req** ∗ *req*)

4.11.3.3 char∗ hreq_hdr (struct **http_req** ∗ *req,* const char ∗ *key*)

4.11.3.4 int hreq_hdr_push (struct **http_req** ∗ *req,* char ∗ *key,* char ∗ *val*)

4.11.3.5 int hreq_is_query (struct **http_req** ∗ *req*)

4.11.3.6 GHashTable∗ hreq_query (struct **http_req** ∗ *req*)

4.11.3.7 void hreq_sign (struct **http_req** ∗ *req,* const char ∗ *bucket,* const char ∗ *key,* char ∗ *b64hmac_out*)

4.11.3.8 bool hstor_add_bucket (struct **hstor_client** ∗ *hstor,* const char ∗ *name*)

4.11.3.9 bool hstor_del (struct **hstor_client** ∗ *hstor,* const char ∗ *bucket,* const char ∗ *key*)

4.11.3.10 bool hstor_del_bucket (struct **hstor_client** ∗ *hstor,* const char ∗ *name*)

4.11.3.11 void hstor_free (struct **hstor_client** ∗ *hstor*)

4.11.3.12 void hstor_free_blist (struct **hstor_blist** ∗ *blist*)

4.11.3.13 void hstor_free_bucket (struct **hstor_bucket** ∗ *buck*)

4.11.3.14 void hstor_free_keylist (struct **hstor_keylist** ∗ *keylist*)

4.11.3.15 void hstor_free_object (struct **hstor_object** ∗ *obj*)

4.11.3.16 bool hstor_get (struct **hstor_client** ∗ *hstor,* const char ∗ *bucket,* const char ∗ *key,* size_t(∗)(void ∗, size_t, size_t, void ∗) *write_cb,* void ∗ *user_data,* bool *want_headers*)

4.11.3.17 void∗ hstor_get_inline (struct **hstor_client** ∗ *hstor,* const char ∗ *bucket,* const char ∗ *key,* bool *want_headers,* size_t ∗ *len*)

4.11.3.18 struct **hstor_keylist**∗ hstor_keys (struct **hstor_client** ∗ *hstor,* const char ∗ *bucket,* const char ∗ *prefix,* const char ∗ *marker,* const char ∗ *delim,* unsigned int *max_keys*)

4.11.3.19 struct **hstor_blist**∗ hstor_list_buckets (struct **hstor_client** ∗ *hstor*)

4.11.3.20 struct **hstor_client**∗ hstor_new (const char ∗ *service_acc,* const char ∗ *service_host,* const char ∗ *user,* const char ∗ *secret_key*)

4.11.3.21 bool hstor_put (struct **hstor_client** ∗ *hstor,* const char ∗ *bucket,* const char ∗ *key,* size_t(∗)(void ∗, size_t, size_t, void ∗) *read_cb,* uint64_t *len,* void ∗ *user_data,* char ∗∗ *user_hdrs*)

4.11.3.22 bool hstor_put_inline (struct **hstor_client** ∗ *hstor,* const char ∗ *bucket,* const char ∗ *key,* void ∗ *data,* uint64_t *len,* char ∗∗ *user_hdrs*)

4.11.3.23 bool hstor_set_format (struct **hstor_client** ∗ *hstor,* enum **hstor_calling_format** *f*)

4.11.3.24 char∗ huri_field_escape (const char ∗ *signed_str,* unsigned char *mask*)

4.11.3.25 int huri_field_unescape (char ∗ *s,* int *s_len*)

4.11.3.26 struct **http_uri**∗ huri_parse (struct **http_uri** ∗ *uri_dest,* char ∗ *uri_src_text*)

4.11.3.27 time_t hutil_str2time (const char ∗ *timestr*)

4.11.3.28 char∗ hutil_time2str (char ∗ *buf,* int *len,* time_t *time*)

4.12 include/ncld.h File Reference

```
#include <stdbool.h>
#include <glib.h>
#include <cldc.h>
```

Data Structures

- struct ncld_sess
- struct ncld_fh
- struct ncld_read

Functions

- struct ncld_sess ∗ ncld_sess_open (const char ∗host, int port, int ∗error, void(∗event)(void ∗, unsigned int), void ∗ev_arg, const char ∗cld_user, const char ∗cld_key, struct hail_log ∗log)
- struct ncld_fh ∗ ncld_open (struct ncld_sess ∗s, const char ∗fname, unsigned int mode, int ∗error, unsigned int events, void(∗event)(void ∗, unsigned int), void ∗ev_arg)
- int ncld_del (struct ncld_sess ∗nsess, const char ∗fname)
- struct ncld_read ∗ ncld_get (struct ncld_fh ∗fh, int ∗error)
- struct ncld_read ∗ ncld_get_meta (struct ncld_fh ∗fh, int ∗error)
- void ncld_read_free (struct ncld_read ∗rp)
- int ncld_write (struct ncld_fh ∗, const void ∗data, long len)
- int ncld_trylock (struct ncld_fh ∗)
- int ncld_qlock (struct ncld_fh ∗)
- int ncld_unlock (struct ncld_fh ∗)
- void ncld_close (struct ncld_fh ∗)
- void ncld_sess_close (struct ncld_sess ∗s)
- void ncld_init (void)

4.12.1 Function Documentation

4.12.1.1 void ncld_close (struct **ncld_fh** ∗)

4.12.1.2 int ncld_del (struct **ncld_sess** ∗ *nsess,* const char ∗ *fname*)

4.12.1.3 struct **ncld_read**∗ ncld_get (struct **ncld_fh** ∗ *fh,* int ∗ *error*)

4.12.1.4 struct **ncld_read**∗ ncld_get_meta (struct **ncld_fh** ∗ *fh,* int ∗ *error*)

4.12.1.5 void ncld_init (void)

4.12.1.6 struct **ncld_fh**∗ ncld_open (struct **ncld_sess** ∗ *s,* const char ∗ *fname,* unsigned int *mode,* int ∗ *error,* unsigned int *events,* void(∗)(void ∗, unsigned int) *event,* void ∗ *ev_arg*)

4.12.1.7 int ncld_qlock (struct **ncld_fh** ∗)

4.12.1.8 void ncld_read_free (struct **ncld_read** ∗ *rp*)

4.12.1.9 void ncld_sess_close (struct **ncld_sess** ∗ *s*)

4.12.1.10 struct **ncld_sess**∗ ncld_sess_open (const char ∗ *host,* int *port,* int ∗ *error,* void(∗)(void ∗, unsigned int) *event,* void ∗ *ev_arg,* const char ∗ *cld_user,* const char ∗ *cld_key,* struct **hail_log** ∗ *log*)

4.12.1.11 int ncld_trylock (struct **ncld_fh** ∗)

4.12.1.12 int ncld_unlock (struct **ncld_fh** ∗)

4.12.1.13 int ncld_write (struct **ncld_fh** ∗ , const void ∗ *data,* long *len*)

4.13 include/objcache.h File Reference

```
#include <glib.h>
#include <stdbool.h>
```

Data Structures

- struct objcache
- struct objcache_entry

Macros

- #define OC_F_DIRTY 0x1
- #define objcache_get(c, k, l) __objcache_get(c, k, l, 0)
- #define objcache_get_dirty(c, k, l) __objcache_get(c, k, l, OC_F_DIRTY)

Functions

- struct objcache_entry ∗ __objcache_get (struct objcache ∗cache, const char ∗key, int klen, unsigned int flag)
- bool objcache_test_dirty (struct objcache ∗cache, struct objcache_entry ∗entry)
- void objcache_put (struct objcache ∗cache, struct objcache_entry ∗entry)
- int objcache_count (struct objcache ∗cache)
- int objcache_init (struct objcache ∗cache)
- void objcache_fini (struct objcache ∗cache)

4.13.1 Macro Definition Documentation

4.13.1.1 #define objcache_get(*c,* *k,* *l*) __objcache_get(c, k, l, 0)

4.13.1.2 #define objcache_get_dirty(*c,* *k,* *l*) __objcache_get(c, k, l, OC_F_DIRTY)

4.13.1.3 #define OC_F_DIRTY 0x1

4.13.2 Function Documentation

4.13.2.1 struct objcache_entry* __objcache_get (struct objcache * *cache,* const char * *key,* int *klen,* unsigned int *flag*)

4.13.2.2 int objcache_count (struct objcache * *cache*)

4.13.2.3 void objcache_fini (struct objcache * *cache*)

4.13.2.4 int objcache_init (struct objcache * *cache*)

4.13.2.5 void objcache_put (struct objcache * *cache,* struct objcache_entry * *entry*)

4.13.2.6 bool objcache_test_dirty (struct objcache * *cache,* struct objcache_entry * *entry*)

Index