

CLD

0.1git

Generated by Doxygen 1.7.1

Thu Oct 7 2010 00:52:21

Contents

1	Data Structure Index	1
1.1	Data Structures	1
2	File Index	3
2.1	File List	3
3	Data Structure Documentation	5
3.1	chunk_check_status Struct Reference	5
3.1.1	Field Documentation	5
3.1.1.1	count	5
3.1.1.2	lastdone	5
3.1.1.3	pad	5
3.1.1.4	state	5
3.2	chunksrv_req Struct Reference	5
3.2.1	Field Documentation	6
3.2.1.1	data_len	6
3.2.1.2	flags	6
3.2.1.3	key_len	6
3.2.1.4	magic	6
3.2.1.5	nonce	6
3.2.1.6	op	6
3.2.1.7	sig	6
3.3	chunksrv_resp Struct Reference	6
3.3.1	Field Documentation	7
3.3.1.1	data_len	7
3.3.1.2	hash	7
3.3.1.3	magic	7
3.3.1.4	nonce	7
3.3.1.5	resp_code	7

3.3.1.6	rsv1	7
3.4	chunksrv_resp_chkstat Struct Reference	7
3.4.1	Field Documentation	7
3.4.1.1	chkstat	7
3.4.1.2	resp	7
3.5	chunksrv_resp_get Struct Reference	7
3.5.1	Field Documentation	8
3.5.1.1	mtime	8
3.5.1.2	resp	8
3.6	cld_dirent_cur Struct Reference	8
3.6.1	Field Documentation	8
3.6.1.1	p	8
3.6.1.2	tmp_len	8
3.7	cld_timer Struct Reference	8
3.7.1	Field Documentation	9
3.7.1.1	cb	9
3.7.1.2	expires	9
3.7.1.3	fired	9
3.7.1.4	name	9
3.7.1.5	on_list	9
3.7.1.6	userdata	9
3.8	cld_timer_list Struct Reference	9
3.8.1	Field Documentation	9
3.8.1.1	list	9
3.8.1.2	runmark	9
3.9	cldc_call_opts Struct Reference	9
3.9.1	Detailed Description	10
3.9.2	Field Documentation	10
3.9.2.1	cb	10
3.9.2.2	private	10
3.9.2.3	resp	10
3.10	cldc_fh Struct Reference	10
3.10.1	Detailed Description	10
3.10.2	Field Documentation	10
3.10.2.1	fh	10
3.10.2.2	sess	10

3.10.2.3	valid	10
3.11	cldc_host Struct Reference	10
3.11.1	Detailed Description	11
3.11.2	Field Documentation	11
3.11.2.1	host	11
3.11.2.2	port	11
3.11.2.3	prio	11
3.11.2.4	weight	11
3.12	cldc_msg Struct Reference	11
3.12.1	Detailed Description	11
3.12.2	Field Documentation	12
3.12.2.1	cb	12
3.12.2.2	cb_private	12
3.12.2.3	copts	12
3.12.2.4	done	12
3.12.2.5	expire_time	12
3.12.2.6	n_pkts	12
3.12.2.7	op	12
3.12.2.8	pkt_info	12
3.12.2.9	sess	12
3.12.2.10	xid	12
3.13	cldc_node_metadata Struct Reference	12
3.13.1	Field Documentation	13
3.13.1.1	flags	13
3.13.1.2	inode_name	13
3.13.1.3	inum	13
3.13.1.4	time_create	13
3.13.1.5	time_modify	13
3.13.1.6	vers	13
3.14	cldc_ops Struct Reference	13
3.14.1	Detailed Description	13
3.14.2	Field Documentation	13
3.14.2.1	event	13
3.14.2.2	pkt_send	13
3.14.2.3	timer_ctl	13
3.15	cldc_pkt_info Struct Reference	14

3.15.1	Field Documentation	14
3.15.1.1	data	14
3.15.1.2	hdr_len	14
3.15.1.3	pkt_len	14
3.15.1.4	retries	14
3.15.1.5	user	14
3.16	cldc_session Struct Reference	14
3.16.1	Detailed Description	15
3.16.2	Field Documentation	16
3.16.2.1	addr	16
3.16.2.2	addr_len	16
3.16.2.3	cfh	16
3.16.2.4	confirmed	16
3.16.2.5	expire_time	16
3.16.2.6	expired	16
3.16.2.7	inode_name_temp	16
3.16.2.8	log	16
3.16.2.9	msg_buf	16
3.16.2.10	msg_buf_len	16
3.16.2.11	msg_buf_op	16
3.16.2.12	msg_scan_time	16
3.16.2.13	next_seqid_in	16
3.16.2.14	next_seqid_in_tr	16
3.16.2.15	next_seqid_out	16
3.16.2.16	ops	16
3.16.2.17	out_msg	16
3.16.2.18	payload	16
3.16.2.19	private	16
3.16.2.20	secret_key	16
3.16.2.21	sid	16
3.16.2.22	user	16
3.17	cldc_udp Struct Reference	17
3.17.1	Detailed Description	17
3.17.2	Field Documentation	17
3.17.2.1	addr	17
3.17.2.2	addr_len	17

3.17.2.3	cb	17
3.17.2.4	cb_private	17
3.17.2.5	fd	17
3.17.2.6	sess	17
3.18	hail_log Struct Reference	17
3.18.1	Field Documentation	18
3.18.1.1	debug	18
3.18.1.2	func	18
3.18.1.3	verbose	18
3.19	hstor_blist Struct Reference	18
3.19.1	Field Documentation	18
3.19.1.1	list	18
3.19.1.2	own_id	18
3.19.1.3	own_name	18
3.20	hstor_bucket Struct Reference	18
3.20.1	Field Documentation	19
3.20.1.1	name	19
3.20.1.2	time_create	19
3.21	hstor_client Struct Reference	19
3.21.1	Field Documentation	19
3.21.1.1	acc	19
3.21.1.2	curl	19
3.21.1.3	host	19
3.21.1.4	key	19
3.21.1.5	user	19
3.21.1.6	verbose	19
3.22	hstor_keylist Struct Reference	19
3.22.1	Field Documentation	20
3.22.1.1	common_pfx	20
3.22.1.2	contents	20
3.22.1.3	delim	20
3.22.1.4	marker	20
3.22.1.5	max_keys	20
3.22.1.6	name	20
3.22.1.7	prefix	20
3.22.1.8	trunc	20

3.23	hstor_object Struct Reference	20
3.23.1	Field Documentation	21
3.23.1.1	etag	21
3.23.1.2	key	21
3.23.1.3	own_id	21
3.23.1.4	own_name	21
3.23.1.5	size	21
3.23.1.6	storage	21
3.23.1.7	time_mod	21
3.24	http_hdr Struct Reference	21
3.24.1	Field Documentation	21
3.24.1.1	key	21
3.24.1.2	val	21
3.25	http_req Struct Reference	21
3.25.1	Field Documentation	22
3.25.1.1	hdr	22
3.25.1.2	major	22
3.25.1.3	method	22
3.25.1.4	minor	22
3.25.1.5	n_hdr	22
3.25.1.6	orig_path	22
3.25.1.7	uri	22
3.26	http_uri Struct Reference	22
3.26.1	Field Documentation	23
3.26.1.1	fragment	23
3.26.1.2	fragment_len	23
3.26.1.3	hostname	23
3.26.1.4	hostname_len	23
3.26.1.5	path	23
3.26.1.6	path_len	23
3.26.1.7	port	23
3.26.1.8	query	23
3.26.1.9	query_len	23
3.26.1.10	scheme	23
3.26.1.11	scheme_len	23
3.26.1.12	userinfo	23

3.26.1.13	userinfo_len	23
3.27	list_head Struct Reference	23
3.27.1	Field Documentation	23
3.27.1.1	next	23
3.27.1.2	prev	23
3.28	ncld_fh Struct Reference	24
3.28.1	Field Documentation	24
3.28.1.1	errc	24
3.28.1.2	event_arg	24
3.28.1.3	event_func	24
3.28.1.4	event_mask	24
3.28.1.5	fh	24
3.28.1.6	is_open	24
3.28.1.7	nios	24
3.28.1.8	sess	24
3.29	ncld_read Struct Reference	24
3.29.1	Field Documentation	25
3.29.1.1	errc	25
3.29.1.2	fh	25
3.29.1.3	is_done	25
3.29.1.4	length	25
3.29.1.5	meta	25
3.29.1.6	ptr	25
3.30	ncld_sess Struct Reference	25
3.30.1	Field Documentation	26
3.30.1.1	cond	26
3.30.1.2	errc	26
3.30.1.3	event	26
3.30.1.4	event_arg	26
3.30.1.5	handles	26
3.30.1.6	host	26
3.30.1.7	is_up	26
3.30.1.8	mutex	26
3.30.1.9	open_done	26
3.30.1.10	port	26
3.30.1.11	thread	26

3.30.1.12 tlist	26
3.30.1.13 to_thread	26
3.30.1.14 udp	26
3.30.1.15 udp_timer	26
3.31 objcache Struct Reference	26
3.31.1 Field Documentation	27
3.31.1.1 lock	27
3.31.1.2 table	27
3.32 objcache_entry Struct Reference	27
3.32.1 Field Documentation	27
3.32.1.1 flags	27
3.32.1.2 hash	27
3.32.1.3 ref	27
3.33 st_client Struct Reference	27
3.33.1 Field Documentation	28
3.33.1.1 fd	28
3.33.1.2 host	28
3.33.1.3 key	28
3.33.1.4 req_buf	28
3.33.1.5 ssl	28
3.33.1.6 ssl_ctx	28
3.33.1.7 user	28
3.33.1.8 verbose	28
3.34 st_keylist Struct Reference	28
3.34.1 Field Documentation	28
3.34.1.1 contents	28
3.34.1.2 name	28
3.35 st_object Struct Reference	28
3.35.1 Field Documentation	29
3.35.1.1 etag	29
3.35.1.2 name	29
3.35.1.3 owner	29
3.35.1.4 size	29
3.35.1.5 time_mod	29
4 File Documentation	31
4.1 include/chunk-private.h File Reference	31

4.1.1	Define Documentation	31
4.1.1.1	BAD_TPATH_FMT	31
4.1.1.2	MDB_TPATH_FMT	31
4.1.1.3	PREFIX_LEN	31
4.2	include/chunk_msg.h File Reference	31
4.2.1	Define Documentation	32
4.2.1.1	CHUNKD_MAGIC	32
4.2.2	Enumeration Type Documentation	32
4.2.2.1	"@0	32
4.2.2.2	chunk_check_state	32
4.2.2.3	chunk_errcode	33
4.2.2.4	chunk_flags	33
4.2.2.5	chunksrv_ops	33
4.3	include/chunkc.h File Reference	34
4.3.1	Function Documentation	36
4.3.1.1	stc_check_start	36
4.3.1.2	stc_check_status	36
4.3.1.3	stc_cp	36
4.3.1.4	stc_del	36
4.3.1.5	stc_free	36
4.3.1.6	stc_free_keylist	36
4.3.1.7	stc_free_object	36
4.3.1.8	stc_get	36
4.3.1.9	stc_get_inline	36
4.3.1.10	stc_get_recv	36
4.3.1.11	stc_get_start	36
4.3.1.12	stc_init	36
4.3.1.13	stc_keys	36
4.3.1.14	stc_new	36
4.3.1.15	stc_ping	36
4.3.1.16	stc_put	36
4.3.1.17	stc_put_inline	36
4.3.1.18	stc_put_send	36
4.3.1.19	stc_put_start	36
4.3.1.20	stc_put_sync	36
4.3.1.21	stc_readport	36

4.3.1.22	stc_table_open	36
4.4	include/chunksrv.h File Reference	36
4.4.1	Function Documentation	37
4.4.1.1	chreq_sign	37
4.4.1.2	req_len	37
4.5	include/cld-private.h File Reference	37
4.6	include/cld_common.h File Reference	37
4.6.1	Define Documentation	38
4.6.1.1	CLD_ALIGN8	38
4.6.1.2	CLD_PKT_FTR_LEN	38
4.6.1.3	PKT_HDR_TO_STR_SCRATCH_LEN	38
4.6.1.4	SIDARG	38
4.6.1.5	SIDFMT	38
4.6.2	Function Documentation	38
4.6.2.1	__attribute__	38
4.6.2.2	__cld_dump_buf	39
4.6.2.3	cld_authcheck	39
4.6.2.4	cld_authsign	39
4.6.2.5	cld_errstr	39
4.6.2.6	cld_opstr	39
4.6.2.7	cld_pkt_hdr_to_str	39
4.6.2.8	cld_rand64	39
4.6.2.9	cld_readport	39
4.6.2.10	cld_sid2llu	39
4.6.2.11	cld_timer_add	39
4.6.2.12	cld_timer_del	39
4.6.2.13	cld_timers_run	39
4.7	include/cldc.h File Reference	39
4.7.1	Function Documentation	42
4.7.1.1	cldc_close	42
4.7.1.2	cldc_copts_get_data	42
4.7.1.3	cldc_copts_get_metadata	42
4.7.1.4	cldc_del	42
4.7.1.5	cldc_dirent_count	42
4.7.1.6	cldc_dirent_cur_fini	42
4.7.1.7	cldc_dirent_cur_init	42

4.7.1.8	<code>cldc_dirent_first</code>	42
4.7.1.9	<code>cldc_dirent_name</code>	42
4.7.1.10	<code>cldc_dirent_next</code>	42
4.7.1.11	<code>cldc_end_sess</code>	42
4.7.1.12	<code>cldc_get</code>	42
4.7.1.13	<code>cldc_getaddr</code>	42
4.7.1.14	<code>cldc_init</code>	42
4.7.1.15	<code>cldc_kill_sess</code>	42
4.7.1.16	<code>cldc_lock</code>	42
4.7.1.17	<code>cldc_new_sess</code>	42
4.7.1.18	<code>cldc_nop</code>	42
4.7.1.19	<code>cldc_open</code>	42
4.7.1.20	<code>cldc_put</code>	42
4.7.1.21	<code>cldc_receive_pkt</code>	42
4.7.1.22	<code>cldc_saveaddr</code>	43
4.7.1.23	<code>cldc_udp_free</code>	43
4.7.1.24	<code>cldc_udp_new</code>	43
4.7.1.25	<code>cldc_udp_pkt_send</code>	43
4.7.1.26	<code>cldc_udp_receive_pkt</code>	43
4.7.1.27	<code>cldc_unlock</code>	43
4.8	<code>include/elist.h</code> File Reference	43
4.8.1	Define Documentation	44
4.8.1.1	<code>INIT_LIST_HEAD</code>	44
4.8.1.2	<code>list_entry</code>	44
4.8.1.3	<code>list_for_each</code>	44
4.8.1.4	<code>list_for_each_entry</code>	44
4.8.1.5	<code>list_for_each_entry_continue</code>	45
4.8.1.6	<code>list_for_each_entry_safe</code>	45
4.8.1.7	<code>list_for_each_prev</code>	45
4.8.1.8	<code>list_for_each_safe</code>	45
4.8.1.9	<code>LIST_HEAD</code>	46
4.8.1.10	<code>LIST_HEAD_INIT</code>	46
4.9	<code>include/hail_log.h</code> File Reference	46
4.9.1	Define Documentation	46
4.9.1.1	<code>ATTR_PRINTF</code>	46
4.9.1.2	<code>HAIL_CRIT</code>	46

4.9.1.3	HAIL_DEBUG	46
4.9.1.4	HAIL_ERR	47
4.9.1.5	HAIL_INFO	47
4.9.1.6	HAIL_VERBOSE	47
4.9.1.7	HAIL_WARN	47
4.10	include/hail_private.h File Reference	47
4.10.1	Function Documentation	47
4.10.1.1	xdr_sizeof	47
4.11	include/hstor.h File Reference	47
4.11.1	Define Documentation	49
4.11.1.1	ARRAY_SIZE	49
4.11.1.2	PATH_ESCAPE_MASK	49
4.11.1.3	QUERY_ESCAPE_MASK	49
4.11.2	Enumeration Type Documentation	49
4.11.2.1	"@1	49
4.11.2.2	ReqACLC	49
4.11.2.3	ReqQ	49
4.11.3	Function Documentation	51
4.11.3.1	hreq_acl_canned	51
4.11.3.2	hreq_free	51
4.11.3.3	hreq_hdr	51
4.11.3.4	hreq_hdr_push	51
4.11.3.5	hreq_is_query	51
4.11.3.6	hreq_query	51
4.11.3.7	hreq_sign	51
4.11.3.8	hstor_add_bucket	51
4.11.3.9	hstor_del	51
4.11.3.10	hstor_del_bucket	51
4.11.3.11	hstor_free	51
4.11.3.12	hstor_free_blist	51
4.11.3.13	hstor_free_bucket	51
4.11.3.14	hstor_free_keylist	51
4.11.3.15	hstor_free_object	51
4.11.3.16	hstor_get	51
4.11.3.17	hstor_get_inline	51
4.11.3.18	hstor_keys	51

4.11.3.19	hstor_list_buckets	51
4.11.3.20	hstor_new	51
4.11.3.21	hstor_put	51
4.11.3.22	hstor_put_inline	51
4.11.3.23	huri_field_escape	51
4.11.3.24	huri_field_unescape	51
4.11.3.25	huri_parse	51
4.11.3.26	hutil_str2time	51
4.11.3.27	hutil_time2str	51
4.12	include/nclld.h File Reference	51
4.12.1	Function Documentation	53
4.12.1.1	nclld_close	53
4.12.1.2	nclld_del	53
4.12.1.3	nclld_get	53
4.12.1.4	nclld_get_meta	53
4.12.1.5	nclld_init	53
4.12.1.6	nclld_open	53
4.12.1.7	nclld_qlock	53
4.12.1.8	nclld_read_free	53
4.12.1.9	nclld_sess_close	53
4.12.1.10	nclld_sess_open	53
4.12.1.11	nclld_trylock	53
4.12.1.12	nclld_unlock	53
4.12.1.13	nclld_write	53
4.13	include/objcache.h File Reference	53
4.13.1	Define Documentation	54
4.13.1.1	objcache_get	54
4.13.1.2	objcache_get_dirty	54
4.13.1.3	OC_F_DIRTY	54
4.13.2	Function Documentation	54
4.13.2.1	__objcache_get	54
4.13.2.2	objcache_count	54
4.13.2.3	objcache_fini	54
4.13.2.4	objcache_init	54
4.13.2.5	objcache_put	54
4.13.2.6	objcache_test_dirty	54

Chapter 1

Data Structure Index

1.1 Data Structures

Here are the data structures with brief descriptions:

chunk_check_status	5
chunksrv_req	5
chunksrv_resp	6
chunksrv_resp_chkstat	7
chunksrv_resp_get	7
cld_dirent_cur	8
cld_timer	8
cld_timer_list	9
cldc_call_opts (Per-operation application options)	9
cldc_fh (Open file handle associated with a session)	10
cldc_host (Information for a single CLD server host)	10
cldc_msg (Outgoing message, from client to server)	11
cldc_node_metadata	12
cldc_ops (Application-supplied facilities)	13
cldc_pkt_info	14
cldc_session (Single CLD client session)	14
cldc_udp (A UDP implementation of the CLD client protocol)	17
hail_log	17
hstor_blist	18
hstor_bucket	18
hstor_client	19
hstor_keylist	19
hstor_object	20
http_hdr	21
http_req	21
http_uri	22
list_head	23
ncld_fh	24
ncld_read	24
ncld_sess	25
objcache	26
objcache_entry	27
st_client	27

st_keylist	28
st_object	28

Chapter 2

File Index

2.1 File List

Here is a list of all files with brief descriptions:

include/chunk-private.h	31
include/chunk_msg.h	31
include/chunkc.h	34
include/chunksrv.h	36
include/cld-private.h	37
include/cld_common.h	37
include/cldc.h	39
include/elist.h	43
include/hail_log.h	46
include/hail_private.h	47
include/hstor.h	47
include/ncl.h	51
include/objcache.h	53

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 [cldc_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]
- 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 `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](#)

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 char* hstor_client::user

3.21.1.6 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 nclد_fh Struct Reference

```
#include <nclد.h>
```

Data Fields

- struct [nclد_sess](#) * [sess](#)
- struct [clدc_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 [nclد_fh::errc](#)

3.28.1.2 void* [nclد_fh::event_arg](#)

3.28.1.3 void(* [nclد_fh::event_func](#))(void *, unsigned int)

3.28.1.4 unsigned int [nclد_fh::event_mask](#)

3.28.1.5 struct [clدc_fh](#)* [nclد_fh::fh](#)

3.28.1.6 bool [nclد_fh::is_open](#)

3.28.1.7 int [nclد_fh::nios](#)

3.28.1.8 struct [nclد_sess](#)* [nclد_fh::sess](#)

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

- include/[nclد.h](#)

3.29 nclد_read Struct Reference

```
#include <nclد.h>
```

Data Fields

- const void * [ptr](#)
- long [length](#)
- struct [clدc_node_metadata](#) [meta](#)
- struct [nclد_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* nclد_sess::cond`
- 3.30.1.2 `int nclد_sess::errc`
- 3.30.1.3 `void(* nclد_sess::event)(void *, unsigned int)`
- 3.30.1.4 `void* nclد_sess::event_arg`
- 3.30.1.5 `GList* nclد_sess::handles`
- 3.30.1.6 `char* nclد_sess::host`
- 3.30.1.7 `bool nclد_sess::is_up`
- 3.30.1.8 `GMutex* nclد_sess::mutex`
- 3.30.1.9 `bool nclد_sess::open_done`
- 3.30.1.10 `unsigned short nclد_sess::port`
- 3.30.1.11 `GThread* nclد_sess::thread`
- 3.30.1.12 `struct cld_timer_list nclد_sess::tlist`
- 3.30.1.13 `int nclد_sess::to_thread[2]`
- 3.30.1.14 `struct cldc_udp* nclد_sess::udp`
- 3.30.1.15 `struct cld_timer nclد_sess::udp_timer`

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

- `include/nclد.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>
```

Defines

- `#define` [MDB_TPATH_FMT](#) "%s/%X"
- `#define` [BAD_TPATH_FMT](#) "%s/bad"
- `#define` [PREFIX_LEN](#) 3

4.1.1 Define 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](#)

Defines

- #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 Define 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)` [read]
- 4.3.1.14 `struct st_client* stc_new (const char * service_host, int port, const char * user, const char * secret_key, bool encrypt)` [read]
- 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

Generated on Thu Oct 7 2010 00:52:21 for CLD by Doxygen

```
#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`

Defines

- #define `CLD_ALIGN8`(n) ((8 - ((n) & 7)) & 7)
- #define `SIDFMT` "%016lX"
- #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 Define 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)) [read]`

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 cld_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)`

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](#)

Defines

- #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 Define 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](#)

Defines

- `#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 Define 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>
```

Functions

- u_long [xdr_sizeof](#) (xdrproc_t, void *)

4.10.1 Function Documentation**4.10.1.1 u_long xdr_sizeof (xdrproc_t, void *)****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](#)

Defines

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

Enumerations

- 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](#) (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_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 Define 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 ReqACLC

Enumerator:

ACLC_PRIV

ACLC_PUB_R

ACLC_PUB_RW

ACLC_AUTH_R

ACLCNUM

4.11.2.3 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) [read]`
- 4.11.3.19 `struct hstor_blist* hstor_list_buckets (struct hstor_client * hstor) [read]`
- 4.11.3.20 `struct hstor_client* hstor_new (const char * service_acc, const char * service_host, const char * user, const char * secret_key) [read]`
- 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 `char* huri_field_escape (char * signed_str, unsigned char mask)`
- 4.11.3.24 `int huri_field_unescape (char * s, int s_len)`
- 4.11.3.25 `struct http_uri* huri_parse (struct http_uri * uri_dest, char * uri_src_text) [read]`

```
#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 `nclد_close` (struct `nclد_fh` *)

4.12.1.2 int `nclد_del` (struct `nclد_sess` * *nsess*, const char * *fname*)

4.12.1.3 struct `nclد_read`* `nclد_get` (struct `nclد_fh` * *fh*, int * *error*) [read]

4.12.1.4 struct `nclد_read`* `nclد_get_meta` (struct `nclد_fh` * *fh*, int * *error*) [read]

4.12.1.5 void `nclد_init` (void)

4.12.1.6 struct `nclد_fh`* `nclد_open` (struct `nclد_sess` * *s*, const char * *fname*, unsigned int *mode*, int * *error*, unsigned int *events*, void(*)(void *, unsigned int) *event*, void * *ev_arg*) [read]

4.12.1.7 int `nclد_qlock` (struct `nclد_fh` *)

4.12.1.8 void `nclد_read_free` (struct `nclد_read` * *rp*)

4.12.1.9 void `nclد_sess_close` (struct `nclد_sess` * *s*)

4.12.1.10 struct `nclد_sess`* `nclد_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*) [read]

4.12.1.11 int `nclد_trylock` (struct `nclد_fh` *)

4.12.1.12 int `nclد_unlock` (struct `nclد_fh` *)

4.12.1.13 int `nclد_write` (struct `nclد_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](#)

Defines

- #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 Define 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)` [`read`]

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

- [__attribute__](#)
 - [cld_common.h](#), 38
 - [__cld_dump_buf](#)
 - [cld_common.h](#), 38
 - [__objcache_get](#)
 - [objcache.h](#), 54
- acc
 - [hstor_client](#), 19
- ACLC_AUTH_R
 - [hstor.h](#), 49
- ACLC_PRIV
 - [hstor.h](#), 49
- ACLC_PUB_R
 - [hstor.h](#), 49
- ACLC_PUB_RW
 - [hstor.h](#), 49
- ACLCNUM
 - [hstor.h](#), 49
- addr
 - [cldc_session](#), 16
 - [cldc_udp](#), 17
- addr_len
 - [cldc_session](#), 16
 - [cldc_udp](#), 17
- ARRAY_SIZE
 - [hstor.h](#), 49
- ATTR_PRINTF
 - [hail_log.h](#), 46
- BAD_TPATH_FMT
 - [chunk-private.h](#), 31
- cb
 - [cld_timer](#), 9
 - [cldc_call_opts](#), 10
 - [cldc_msg](#), 12
 - [cldc_udp](#), 17
- cb_private
 - [cldc_msg](#), 12
 - [cldc_udp](#), 17
- cfh
 - [cldc_session](#), 16
- CHD_CSUM_SZ
 - [chunk_msg.h](#), 32
- CHD_KEY_SZ
 - [chunk_msg.h](#), 32
- CHD_MAGIC_SZ
 - [chunk_msg.h](#), 32
- CHD_SIG_SZ
 - [chunk_msg.h](#), 32
- CHD_USER_SZ
 - [chunk_msg.h](#), 32
- che_AccessDenied
 - [chunk_msg.h](#), 33
- che_Busy
 - [chunk_msg.h](#), 33
- che_InternalError
 - [chunk_msg.h](#), 33
- che_InvalidArgument
 - [chunk_msg.h](#), 33
- che_InvalidKey
 - [chunk_msg.h](#), 33
- che_InvalidTable
 - [chunk_msg.h](#), 33
- che_InvalidURI
 - [chunk_msg.h](#), 33
- che_KeyExists
 - [chunk_msg.h](#), 33
- che_NoSuchKey
 - [chunk_msg.h](#), 33
- che_SignatureDoesNotMatch
 - [chunk_msg.h](#), 33
- che_Success
 - [chunk_msg.h](#), 33
- CHF_SYNC
 - [chunk_msg.h](#), 33
- CHF_TBL_CREAT
 - [chunk_msg.h](#), 33
- CHF_TBL_EXCL
 - [chunk_msg.h](#), 33
- chk_Active
 - [chunk_msg.h](#), 32
- chk_Idle
 - [chunk_msg.h](#), 32
- chk_Off
 - [chunk_msg.h](#), 32
- chkstat
 - [chunksrv_resp_chkstat](#), 7
- CHO_CHECK_START

- chunk_msg.h, 33
- CHO_CHECK_STATUS
 - chunk_msg.h, 33
- CHO_CP
 - chunk_msg.h, 33
- CHO_DEL
 - chunk_msg.h, 33
- CHO_GET
 - chunk_msg.h, 33
- CHO_GET_META
 - chunk_msg.h, 33
- CHO_LIST
 - chunk_msg.h, 33
- CHO_LOGIN
 - chunk_msg.h, 33
- CHO_NOP
 - chunk_msg.h, 33
- CHO_PUT
 - chunk_msg.h, 33
- CHO_START_TLS
 - chunk_msg.h, 33
- CHO_TABLE_OPEN
 - chunk_msg.h, 33
- chreq_sign
 - chunksrv.h, 37
- chunk-private.h
 - BAD_TPATH_FMT, 31
 - MDB_TPATH_FMT, 31
 - PREFIX_LEN, 31
- chunk_msg.h
 - CHD_CSUM_SZ, 32
 - CHD_KEY_SZ, 32
 - CHD_MAGIC_SZ, 32
 - CHD_SIG_SZ, 32
 - CHD_USER_SZ, 32
 - che_AccessDenied, 33
 - che_Busy, 33
 - che_InternalError, 33
 - che_InvalidArgument, 33
 - che_InvalidKey, 33
 - che_InvalidTable, 33
 - che_InvalidURI, 33
 - che_KeyExists, 33
 - che_NoSuchKey, 33
 - che_SignatureDoesNotMatch, 33
 - che_Success, 33
 - CHF_SYNC, 33
 - CHF_TBL_CREAT, 33
 - CHF_TBL_EXCL, 33
 - chk_Active, 32
 - chk_Idle, 32
 - chk_Off, 32
 - CHO_CHECK_START, 33
 - CHO_CHECK_STATUS, 33
 - CHO_CP, 33
 - CHO_DEL, 33
 - CHO_GET, 33
 - CHO_GET_META, 33
 - CHO_LIST, 33
 - CHO_LOGIN, 33
 - CHO_NOP, 33
 - CHO_PUT, 33
 - CHO_START_TLS, 33
 - CHO_TABLE_OPEN, 33
 - chunk_check_state
 - chunk_msg.h, 32
 - chunk_check_status, 5
 - count, 5
 - lastdone, 5
 - pad, 5
 - state, 5
 - chunk_errcode
 - chunk_msg.h, 32
 - chunk_flags
 - chunk_msg.h, 33
 - chunk_msg.h
 - chunk_check_state, 32
 - chunk_errcode, 32
 - chunk_flags, 33
 - CHUNKD_MAGIC, 32
 - chunksrv_ops, 33
 - chunkc.h
 - stc_check_start, 36
 - stc_check_status, 36
 - stc_cp, 36
 - stc_del, 36
 - stc_free, 36
 - stc_free_keylist, 36
 - stc_free_object, 36
 - stc_get, 36
 - stc_get_inline, 36
 - stc_get_recv, 36
 - stc_get_start, 36
 - stc_init, 36
 - stc_keys, 36
 - stc_new, 36
 - stc_ping, 36
 - stc_put, 36
 - stc_put_inline, 36
 - stc_put_send, 36
 - stc_put_start, 36
 - stc_put_sync, 36
 - stc_readport, 36
 - stc_table_open, 36
 - CHUNKD_MAGIC
 - chunk_msg.h, 32
 - chunksrv.h
 - chreq_sign, 37

- req_len, 37
- chunksrv_ops
 - chunk_msg.h, 33
- chunksrv_req, 5
 - data_len, 6
 - flags, 6
 - key_len, 6
 - magic, 6
 - nonce, 6
 - op, 6
 - sig, 6
- chunksrv_resp, 6
 - data_len, 7
 - hash, 7
 - magic, 7
 - nonce, 7
 - resp_code, 7
 - rsv1, 7
- chunksrv_resp_chkstat, 7
 - chkstat, 7
 - resp, 7
- chunksrv_resp_get, 7
 - mtime, 8
 - resp, 8
- CLD_ALIGN8
 - cld_common.h, 38
- cld_authcheck
 - cld_common.h, 39
- cld_authsign
 - cld_common.h, 39
- cld_common.h
 - __attribute__, 38
 - __cld_dump_buf, 38
 - CLD_ALIGN8, 38
 - cld_authcheck, 39
 - cld_authsign, 39
 - cld_errstr, 39
 - cld_opstr, 39
 - CLD_PKT_FTR_LEN, 38
 - cld_pkt_hdr_to_str, 39
 - cld_rand64, 39
 - cld_readport, 39
 - cld_sid2llu, 39
 - cld_timer_add, 39
 - cld_timer_del, 39
 - cld_timers_run, 39
 - PKT_HDR_TO_STR_SCRATCH_LEN, 38
 - SIDARG, 38
 - SIDFMT, 38
- cld_dirent_cur, 8
 - p, 8
 - tmp_len, 8
- cld_errstr
 - cld_common.h, 39
- cld_opstr
 - cld_common.h, 39
- CLD_PKT_FTR_LEN
 - cld_common.h, 38
- cld_pkt_hdr_to_str
 - cld_common.h, 39
- cld_rand64
 - cld_common.h, 39
- cld_readport
 - cld_common.h, 39
- cld_sid2llu
 - cld_common.h, 39
- cld_timer, 8
 - cb, 9
 - expires, 9
 - fired, 9
 - name, 9
 - on_list, 9
 - userdata, 9
- cld_timer_add
 - cld_common.h, 39
- cld_timer_del
 - cld_common.h, 39
- cld_timer_list, 9
 - list, 9
 - runmark, 9
- cld_timers_run
 - cld_common.h, 39
- cldc.h
 - cldc_close, 42
 - cldc_copts_get_data, 42
 - cldc_copts_get_metadata, 42
 - cldc_del, 42
 - cldc_dirent_count, 42
 - cldc_dirent_cur_fini, 42
 - cldc_dirent_cur_init, 42
 - cldc_dirent_first, 42
 - cldc_dirent_name, 42
 - cldc_dirent_next, 42
 - cldc_end_sess, 42
 - cldc_get, 42
 - cldc_getaddr, 42
 - cldc_init, 42
 - cldc_kill_sess, 42
 - cldc_lock, 42
 - cldc_new_sess, 42
 - cldc_nop, 42
 - cldc_open, 42
 - cldc_put, 42
 - cldc_receive_pkt, 42
 - cldc_saveaddr, 43
 - cldc_udp_free, 43
 - cldc_udp_new, 43
 - cldc_udp_pkt_send, 43

- cldc_udp_receive_pkt, 43
 - cldc_unlock, 43
- cldc_call_opts, 9
 - cb, 10
 - private, 10
 - resp, 10
- cldc_close
 - cldc.h, 42
- cldc_copts_get_data
 - cldc.h, 42
- cldc_copts_get_metadata
 - cldc.h, 42
- cldc_del
 - cldc.h, 42
- cldc_dirent_count
 - cldc.h, 42
- cldc_dirent_cur_fini
 - cldc.h, 42
- cldc_dirent_cur_init
 - cldc.h, 42
- cldc_dirent_first
 - cldc.h, 42
- cldc_dirent_name
 - cldc.h, 42
- cldc_dirent_next
 - cldc.h, 42
- cldc_end_sess
 - cldc.h, 42
- cldc_fh, 10
 - fh, 10
 - sess, 10
 - valid, 10
- cldc_get
 - cldc.h, 42
- cldc_getaddr
 - cldc.h, 42
- cldc_host, 10
 - host, 11
 - port, 11
 - prio, 11
 - weight, 11
- cldc_init
 - cldc.h, 42
- cldc_kill_sess
 - cldc.h, 42
- cldc_lock
 - cldc.h, 42
- cldc_msg, 11
 - cb, 12
 - cb_private, 12
 - copts, 12
 - done, 12
 - expire_time, 12
 - n_pkts, 12
 - op, 12
 - pkt_info, 12
 - sess, 12
 - xid, 12
- cldc_new_sess
 - cldc.h, 42
- cldc_node_metadata, 12
 - flags, 13
 - inode_name, 13
 - inum, 13
 - time_create, 13
 - time_modify, 13
 - vers, 13
- cldc_nop
 - cldc.h, 42
- cldc_open
 - cldc.h, 42
- cldc_ops, 13
 - event, 13
 - pkt_send, 13
 - timer_ctl, 13
- cldc_pkt_info, 14
 - data, 14
 - hdr_len, 14
 - pkt_len, 14
 - retries, 14
 - user, 14
- cldc_put
 - cldc.h, 42
- cldc_receive_pkt
 - cldc.h, 42
- cldc_saveaddr
 - cldc.h, 43
- cldc_session, 14
 - addr, 16
 - addr_len, 16
 - cfh, 16
 - confirmed, 16
 - expire_time, 16
 - expired, 16
 - inode_name_temp, 16
 - log, 16
 - msg_buf, 16
 - msg_buf_len, 16
 - msg_buf_op, 16
 - msg_scan_time, 16
 - next_seqid_in, 16
 - next_seqid_in_tr, 16
 - next_seqid_out, 16
 - ops, 16
 - out_msg, 16
 - payload, 16
 - private, 16
 - secret_key, 16

- sid, 16
- user, 16
- cldc_udp, 17
 - addr, 17
 - addr_len, 17
 - cb, 17
 - cb_private, 17
 - fd, 17
 - sess, 17
- cldc_udp_free
 - cldc.h, 43
- cldc_udp_new
 - cldc.h, 43
- cldc_udp_pkt_send
 - cldc.h, 43
- cldc_udp_receive_pkt
 - cldc.h, 43
- cldc_unlock
 - cldc.h, 43
- common_pfx
 - hstor_keylist, 20
- cond
 - nclد_sess, 26
- confirmed
 - cldc_session, 16
- contents
 - hstor_keylist, 20
 - st_keylist, 28
- copts
 - cldc_msg, 12
- count
 - chunk_check_status, 5
- curl
 - hstor_client, 19
- data
 - cldc_pkt_info, 14
- data_len
 - chunksrv_req, 6
 - chunksrv_resp, 7
- debug
 - hail_log, 18
- delim
 - hstor_keylist, 20
- done
 - cldc_msg, 12
- elist.h
 - INIT_LIST_HEAD, 44
 - list_entry, 44
 - list_for_each, 44
 - list_for_each_entry, 44
 - list_for_each_entry_continue, 44
 - list_for_each_entry_safe, 45
 - list_for_each_prev, 45
 - list_for_each_safe, 45
 - LIST_HEAD, 45
 - LIST_HEAD_INIT, 46
- errc
 - nclد_fh, 24
 - nclد_read, 25
 - nclد_sess, 26
- etag
 - hstor_object, 21
 - st_object, 29
- event
 - cldc_ops, 13
 - nclد_sess, 26
- event_arg
 - nclد_fh, 24
 - nclد_sess, 26
- event_func
 - nclد_fh, 24
- event_mask
 - nclد_fh, 24
- expire_time
 - cldc_msg, 12
 - cldc_session, 16
- expired
 - cldc_session, 16
- expires
 - cld_timer, 9
- fd
 - cldc_udp, 17
 - st_client, 28
- fh
 - cldc_fh, 10
 - nclد_fh, 24
 - nclد_read, 25
- fired
 - cld_timer, 9
- flags
 - chunksrv_req, 6
 - cldc_node_metadata, 13
 - objcache_entry, 27
- fragment
 - http_uri, 23
- fragment_len
 - http_uri, 23
- func
 - hail_log, 18
- HAIL_CRIT
 - hail_log.h, 46
- HAIL_DEBUG
 - hail_log.h, 46
- HAIL_ERR

- hail_log.h, 46
- HAIL_INFO
 - hail_log.h, 47
- hail_log, 17
 - debug, 18
 - func, 18
 - verbose, 18
- hail_log.h
 - ATTR_PRINTF, 46
 - HAIL_CRIT, 46
 - HAIL_DEBUG, 46
 - HAIL_ERR, 46
 - HAIL_INFO, 47
 - HAIL_VERBOSE, 47
 - HAIL_WARN, 47
- hail_private.h
 - xdr_sizeof, 47
- HAIL_VERBOSE
 - hail_log.h, 47
- HAIL_WARN
 - hail_log.h, 47
- handles
 - ncld_sess, 26
- hash
 - chunksrv_resp, 7
 - objcache_entry, 27
- hdr
 - http_req, 22
- hdr_len
 - cldc_pkt_info, 14
- host
 - cldc_host, 11
 - hstor_client, 19
 - ncld_sess, 26
 - st_client, 28
- hostname
 - http_uri, 23
- hostname_len
 - http_uri, 23
- HREQ_MAX_HDR
 - hstor.h, 49
- hreq_acl_canned
 - hstor.h, 51
- hreq_free
 - hstor.h, 51
- hreq_hdr
 - hstor.h, 51
- hreq_hdr_push
 - hstor.h, 51
- hreq_is_query
 - hstor.h, 51
- hreq_query
 - hstor.h, 51
- hreq_sign
 - hstor.h, 51
- hstor.h, 51
- hstor.h
 - ACLC_AUTH_R, 49
 - ACLC_PRIV, 49
 - ACLC_PUB_R, 49
 - ACLC_PUB_RW, 49
 - ACLCNUM, 49
 - ARRAY_SIZE, 49
 - HREQ_MAX_HDR, 49
 - hreq_acl_canned, 51
 - hreq_free, 51
 - hreq_hdr, 51
 - hreq_hdr_push, 51
 - hreq_is_query, 51
 - hreq_query, 51
 - hreq_sign, 51
 - hstor_add_bucket, 51
 - hstor_del, 51
 - hstor_del_bucket, 51
 - hstor_free, 51
 - hstor_free_blist, 51
 - hstor_free_bucket, 51
 - hstor_free_keylist, 51
 - hstor_free_object, 51
 - hstor_get, 51
 - hstor_get_inline, 51
 - hstor_keys, 51
 - hstor_list_buckets, 51
 - hstor_new, 51
 - hstor_put, 51
 - hstor_put_inline, 51
 - huri_field_escape, 51
 - huri_field_unescape, 51
 - huri_parse, 51
 - hutil_str2time, 51
 - hutil_time2str, 51
 - PATH_ESCAPE_MASK, 49
 - QUERY_ESCAPE_MASK, 49
 - ReqACLC, 49
 - ReqQ, 49
 - URIQ_ACL, 49
 - URIQ_LOCATION, 49
 - URIQ_LOGGING, 49
 - URIQ_TORRENT, 49
 - URIQNUM, 49
- hstor_add_bucket
 - hstor.h, 51
- hstor_blist, 18
 - list, 18
 - own_id, 18
 - own_name, 18
- hstor_bucket, 18
 - name, 19
 - time_create, 19

- hstor_client, 19
 - acc, 19
 - curl, 19
 - host, 19
 - key, 19
 - user, 19
 - verbose, 19
- hstor_del
 - hstor.h, 51
- hstor_del_bucket
 - hstor.h, 51
- hstor_free
 - hstor.h, 51
- hstor_free_blist
 - hstor.h, 51
- hstor_free_bucket
 - hstor.h, 51
- hstor_free_keylist
 - hstor.h, 51
- hstor_free_object
 - hstor.h, 51
- hstor_get
 - hstor.h, 51
- hstor_get_inline
 - hstor.h, 51
- hstor_keylist, 19
 - common_pfx, 20
 - contents, 20
 - delim, 20
 - marker, 20
 - max_keys, 20
 - name, 20
 - prefix, 20
 - trunc, 20
- hstor_keys
 - hstor.h, 51
- hstor_list_buckets
 - hstor.h, 51
- hstor_new
 - hstor.h, 51
- hstor_object, 20
 - etag, 21
 - key, 21
 - own_id, 21
 - own_name, 21
 - size, 21
 - storage, 21
 - time_mod, 21
- hstor_put
 - hstor.h, 51
- hstor_put_inline
 - hstor.h, 51
- http_hdr, 21
 - key, 21
 - val, 21
- http_req, 21
 - hdr, 22
 - major, 22
 - method, 22
 - minor, 22
 - n_hdr, 22
 - orig_path, 22
 - uri, 22
- http_uri, 22
 - fragment, 23
 - fragment_len, 23
 - hostname, 23
 - hostname_len, 23
 - path, 23
 - path_len, 23
 - port, 23
 - query, 23
 - query_len, 23
 - scheme, 23
 - scheme_len, 23
 - userinfo, 23
 - userinfo_len, 23
- huri_field_escape
 - hstor.h, 51
- huri_field_unescape
 - hstor.h, 51
- huri_parse
 - hstor.h, 51
- hutil_str2time
 - hstor.h, 51
- hutil_time2str
 - hstor.h, 51
- include/chunk-private.h, 31
- include/chunk_msg.h, 31
- include/chunkc.h, 34
- include/chunksrv.h, 36
- include/cld-private.h, 37
- include/cld_common.h, 37
- include/cldc.h, 39
- include/elist.h, 43
- include/hail_log.h, 46
- include/hail_private.h, 47
- include/hstor.h, 47
- include/nclld.h, 51
- include/objcache.h, 53
- INIT_LIST_HEAD
 - elist.h, 44
- inode_name
 - cldc_node_metadata, 13
- inode_name_temp
 - cldc_session, 16
- inum

- cldc_node_metadata, 13
- is_done
 - ncld_read, 25
- is_open
 - ncld_fh, 24
- is_up
 - ncld_sess, 26
- key
 - hstor_client, 19
 - hstor_object, 21
 - http_hdr, 21
 - st_client, 28
- key_len
 - chunksrv_req, 6
- lastdone
 - chunk_check_status, 5
- length
 - ncld_read, 25
- list
 - cld_timer_list, 9
 - hstor_blist, 18
- list_entry
 - elist.h, 44
- list_for_each
 - elist.h, 44
- list_for_each_entry
 - elist.h, 44
- list_for_each_entry_continue
 - elist.h, 44
- list_for_each_entry_safe
 - elist.h, 45
- list_for_each_prev
 - elist.h, 45
- list_for_each_safe
 - elist.h, 45
- LIST_HEAD
 - elist.h, 45
- list_head, 23
 - next, 23
 - prev, 23
- LIST_HEAD_INIT
 - elist.h, 46
- lock
 - objcache, 27
- log
 - cldc_session, 16
- magic
 - chunksrv_req, 6
 - chunksrv_resp, 7
- major
 - http_req, 22
- marker
 - hstor_keylist, 20
- max_keys
 - hstor_keylist, 20
- MDB_TPATH_FMT
 - chunk-private.h, 31
- meta
 - ncld_read, 25
- method
 - http_req, 22
- minor
 - http_req, 22
- msg_buf
 - cldc_session, 16
- msg_buf_len
 - cldc_session, 16
- msg_buf_op
 - cldc_session, 16
- msg_scan_time
 - cldc_session, 16
- mtime
 - chunksrv_resp_get, 8
- mutex
 - ncld_sess, 26
- n_hdr
 - http_req, 22
- n_pkts
 - cldc_msg, 12
- name
 - cld_timer, 9
 - hstor_bucket, 19
 - hstor_keylist, 20
 - st_keylist, 28
 - st_object, 29
- ncld.h
 - ncld_close, 53
 - ncld_del, 53
 - ncld_get, 53
 - ncld_get_meta, 53
 - ncld_init, 53
 - ncld_open, 53
 - ncld_qlock, 53
 - ncld_read_free, 53
 - ncld_sess_close, 53
 - ncld_sess_open, 53
 - ncld_trylock, 53
 - ncld_unlock, 53
 - ncld_write, 53
- ncld_close
 - ncld.h, 53
- ncld_del
 - ncld.h, 53
- ncld_fh, 24

- errc, 24
- event_arg, 24
- event_func, 24
- event_mask, 24
- fh, 24
- is_open, 24
- nios, 24
- sess, 24
- nclد_get
 - nclد.h, 53
- nclد_get_meta
 - nclد.h, 53
- nclد_init
 - nclد.h, 53
- nclد_open
 - nclد.h, 53
- nclد_qlock
 - nclد.h, 53
- nclد_read, 24
 - errc, 25
 - fh, 25
 - is_done, 25
 - length, 25
 - meta, 25
 - ptr, 25
- nclد_read_free
 - nclد.h, 53
- nclد_sess, 25
 - cond, 26
 - errc, 26
 - event, 26
 - event_arg, 26
 - handles, 26
 - host, 26
 - is_up, 26
 - mutex, 26
 - open_done, 26
 - port, 26
 - thread, 26
 - tlist, 26
 - to_thread, 26
 - udp, 26
 - udp_timer, 26
- nclد_sess_close
 - nclد.h, 53
- nclد_sess_open
 - nclد.h, 53
- nclد_trylock
 - nclد.h, 53
- nclد_unlock
 - nclد.h, 53
- nclد_write
 - nclد.h, 53
- next
 - list_head, 23
- next_seqid_in
 - cldc_session, 16
- next_seqid_in_tr
 - cldc_session, 16
- next_seqid_out
 - cldc_session, 16
- nios
 - nclد_fh, 24
- nonce
 - chunksrv_req, 6
 - chunksrv_resp, 7
- objcache, 26
 - lock, 27
 - table, 27
- objcache.h
 - __objcache_get, 54
 - objcache_count, 54
 - objcache_fini, 54
 - objcache_get, 54
 - objcache_get_dirty, 54
 - objcache_init, 54
 - objcache_put, 54
 - objcache_test_dirty, 54
 - OC_F_DIRTY, 54
- objcache_count
 - objcache.h, 54
- objcache_entry, 27
 - flags, 27
 - hash, 27
 - ref, 27
- objcache_fini
 - objcache.h, 54
- objcache_get
 - objcache.h, 54
- objcache_get_dirty
 - objcache.h, 54
- objcache_init
 - objcache.h, 54
- objcache_put
 - objcache.h, 54
- objcache_test_dirty
 - objcache.h, 54
- OC_F_DIRTY
 - objcache.h, 54
- on_list
 - cld_timer, 9
- op
 - chunksrv_req, 6
 - cldc_msg, 12
- open_done
 - nclد_sess, 26
- ops

- cldc_session, 16
- orig_path
 - http_req, 22
- out_msg
 - cldc_session, 16
- own_id
 - hstor_blist, 18
 - hstor_object, 21
- own_name
 - hstor_blist, 18
 - hstor_object, 21
- owner
 - st_object, 29
- p
 - cld_dirent_cur, 8
- pad
 - chunk_check_status, 5
- path
 - http_uri, 23
- PATH_ESCAPE_MASK
 - hstor.h, 49
- path_len
 - http_uri, 23
- payload
 - cldc_session, 16
- PKT_HDR_TO_STR_SCRATCH_LEN
 - cld_common.h, 38
- pkt_info
 - cldc_msg, 12
- pkt_len
 - cldc_pkt_info, 14
- pkt_send
 - cldc_ops, 13
- port
 - cldc_host, 11
 - http_uri, 23
 - ncld_sess, 26
- prefix
 - hstor_keylist, 20
- PREFIX_LEN
 - chunk-private.h, 31
- prev
 - list_head, 23
- prio
 - cldc_host, 11
- private
 - cldc_call_opts, 10
 - cldc_session, 16
- ptr
 - ncld_read, 25
- query
 - http_uri, 23
- QUERY_ESCAPE_MASK
 - hstor.h, 49
- query_len
 - http_uri, 23
- ref
 - objcache_entry, 27
- req_buf
 - st_client, 28
- req_len
 - chunksrv.h, 37
- ReqACLC
 - hstor.h, 49
- ReqQ
 - hstor.h, 49
- resp
 - chunksrv_resp_chkstat, 7
 - chunksrv_resp_get, 8
 - cldc_call_opts, 10
- resp_code
 - chunksrv_resp, 7
- retries
 - cldc_pkt_info, 14
- rsvl
 - chunksrv_resp, 7
- runmark
 - cld_timer_list, 9
- scheme
 - http_uri, 23
- scheme_len
 - http_uri, 23
- secret_key
 - cldc_session, 16
- sess
 - cldc_fh, 10
 - cldc_msg, 12
 - cldc_udp, 17
 - ncld_fh, 24
- sid
 - cldc_session, 16
- SIDARG
 - cld_common.h, 38
- SIDFMT
 - cld_common.h, 38
- sig
 - chunksrv_req, 6
- size
 - hstor_object, 21
 - st_object, 29
- ssl
 - st_client, 28
- ssl_ctx
 - st_client, 28

- st_client, 27
 - fd, 28
 - host, 28
 - key, 28
 - req_buf, 28
 - ssl, 28
 - ssl_ctx, 28
 - user, 28
 - verbose, 28
- st_keylist, 28
 - contents, 28
 - name, 28
- st_object, 28
 - etag, 29
 - name, 29
 - owner, 29
 - size, 29
 - time_mod, 29
- state
 - chunk_check_status, 5
- stc_check_start
 - chunkc.h, 36
- stc_check_status
 - chunkc.h, 36
- stc_cp
 - chunkc.h, 36
- stc_del
 - chunkc.h, 36
- stc_free
 - chunkc.h, 36
- stc_free_keylist
 - chunkc.h, 36
- stc_free_object
 - chunkc.h, 36
- stc_get
 - chunkc.h, 36
- stc_get_inline
 - chunkc.h, 36
- stc_get_recv
 - chunkc.h, 36
- stc_get_start
 - chunkc.h, 36
- stc_init
 - chunkc.h, 36
- stc_keys
 - chunkc.h, 36
- stc_new
 - chunkc.h, 36
- stc_ping
 - chunkc.h, 36
- stc_put
 - chunkc.h, 36
- stc_put_inline
 - chunkc.h, 36
- stc_put_send
 - chunkc.h, 36
- stc_put_start
 - chunkc.h, 36
- stc_put_sync
 - chunkc.h, 36
- stc_readport
 - chunkc.h, 36
- stc_table_open
 - chunkc.h, 36
- storage
 - hstor_object, 21
- table
 - objcache, 27
- thread
 - ncld_sess, 26
- time_create
 - cldc_node_metadata, 13
 - hstor_bucket, 19
- time_mod
 - hstor_object, 21
 - st_object, 29
- time_modify
 - cldc_node_metadata, 13
- timer_ctl
 - cldc_ops, 13
- tlist
 - ncld_sess, 26
- tmp_len
 - cld_dirent_cur, 8
- to_thread
 - ncld_sess, 26
- trunc
 - hstor_keylist, 20
- udp
 - ncld_sess, 26
- udp_timer
 - ncld_sess, 26
- uri
 - http_req, 22
- URIQ_ACL
 - hstor.h, 49
- URIQ_LOCATION
 - hstor.h, 49
- URIQ_LOGGING
 - hstor.h, 49
- URIQ_TORRENT
 - hstor.h, 49
- URIQNUM
 - hstor.h, 49
- user
 - cldc_pkt_info, 14

- cldc_session, [16](#)
 - hstor_client, [19](#)
 - st_client, [28](#)
- userdata
 - cld_timer, [9](#)
- userinfo
 - http_uri, [23](#)
- userinfo_len
 - http_uri, [23](#)
- val
 - http_hdr, [21](#)
- valid
 - cldc_fh, [10](#)
- verbose
 - hail_log, [18](#)
 - hstor_client, [19](#)
 - st_client, [28](#)
- vers
 - cldc_node_metadata, [13](#)
- weight
 - cldc_host, [11](#)
- xdr_sizeof
 - hail_private.h, [47](#)
- xid
 - cldc_msg, [12](#)