

CLD

0.1git

Generated by Doxygen 1.6.1

Tue Apr 5 18:17:20 2011

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	6
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	7
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	8
3.4.1	Field Documentation	8
3.4.1.1	chkstat	8
3.4.1.2	resp	8
3.5	chunksrv_resp_get Struct Reference	9
3.5.1	Field Documentation	9
3.5.1.1	mtime	9
3.5.1.2	resp	9
3.6	cld_dirent_cur Struct Reference	10
3.6.1	Field Documentation	10
3.6.1.1	p	10
3.6.1.2	tmp_len	10
3.7	cld_timer Struct Reference	11
3.7.1	Field Documentation	11
3.7.1.1	cb	11
3.7.1.2	expires	11
3.7.1.3	fired	11
3.7.1.4	name	11
3.7.1.5	on_list	11
3.7.1.6	userdata	11
3.8	cld_timer_list Struct Reference	12
3.8.1	Field Documentation	12
3.8.1.1	list	12
3.8.1.2	runmark	12
3.9	cldc_call_opts Struct Reference	13
3.9.1	Detailed Description	13
3.9.2	Field Documentation	13
3.9.2.1	cb	13
3.9.2.2	private	13
3.9.2.3	resp	13
3.10	cldc_fh Struct Reference	14
3.10.1	Detailed Description	14
3.10.2	Field Documentation	14
3.10.2.1	fh	14
3.10.2.2	sess	14

3.10.2.3	valid	14
3.11	cldc_host Struct Reference	15
3.11.1	Detailed Description	15
3.11.2	Field Documentation	15
3.11.2.1	host	15
3.11.2.2	port	15
3.11.2.3	prio	15
3.11.2.4	weight	15
3.12	cldc_msg Struct Reference	16
3.12.1	Detailed Description	16
3.12.2	Field Documentation	16
3.12.2.1	cb	16
3.12.2.2	cb_private	16
3.12.2.3	copts	16
3.12.2.4	done	16
3.12.2.5	expire_time	16
3.12.2.6	n_pkts	16
3.12.2.7	op	16
3.12.2.8	pkt_info	16
3.12.2.9	sess	16
3.12.2.10	xid	16
3.13	cldc_node_metadata Struct Reference	17
3.13.1	Field Documentation	17
3.13.1.1	flags	17
3.13.1.2	inode_name	17
3.13.1.3	inum	17
3.13.1.4	time_create	17
3.13.1.5	time_modify	17
3.13.1.6	vers	17
3.14	cldc_ops Struct Reference	18
3.14.1	Detailed Description	18
3.14.2	Field Documentation	18
3.14.2.1	event	18
3.14.2.2	pkt_send	18
3.14.2.3	timer_ctl	18
3.15	cldc_pkt_info Struct Reference	19

3.15.1	Field Documentation	19
3.15.1.1	data	19
3.15.1.2	hdr_len	19
3.15.1.3	pkt_len	19
3.15.1.4	retries	19
3.15.1.5	user	19
3.16	cldc_session Struct Reference	20
3.16.1	Detailed Description	20
3.16.2	Field Documentation	21
3.16.2.1	addr	21
3.16.2.2	addr_len	21
3.16.2.3	cfh	21
3.16.2.4	confirmed	21
3.16.2.5	expire_time	21
3.16.2.6	expired	21
3.16.2.7	inode_name_temp	21
3.16.2.8	log	21
3.16.2.9	msg_buf	21
3.16.2.10	msg_buf_len	21
3.16.2.11	msg_buf_op	21
3.16.2.12	msg_scan_time	21
3.16.2.13	next_seqid_in	21
3.16.2.14	next_seqid_in_tr	21
3.16.2.15	next_seqid_out	21
3.16.2.16	ops	21
3.16.2.17	out_msg	21
3.16.2.18	payload	21
3.16.2.19	private	21
3.16.2.20	secret_key	21
3.16.2.21	sid	21
3.16.2.22	user	21
3.17	cldc_udp Struct Reference	22
3.17.1	Detailed Description	22
3.17.2	Field Documentation	22
3.17.2.1	addr	22
3.17.2.2	addr_len	22

3.17.2.3	cb	22
3.17.2.4	cb_private	22
3.17.2.5	fd	22
3.17.2.6	sess	22
3.18	hail_log Struct Reference	23
3.18.1	Field Documentation	23
3.18.1.1	debug	23
3.18.1.2	func	23
3.18.1.3	verbose	23
3.19	hstor_blist Struct Reference	24
3.19.1	Field Documentation	24
3.19.1.1	list	24
3.19.1.2	own_id	24
3.19.1.3	own_name	24
3.20	hstor_bucket Struct Reference	25
3.20.1	Field Documentation	25
3.20.1.1	name	25
3.20.1.2	time_create	25
3.21	hstor_client Struct Reference	26
3.21.1	Field Documentation	26
3.21.1.1	acc	26
3.21.1.2	curl	26
3.21.1.3	host	26
3.21.1.4	key	26
3.21.1.5	user	26
3.21.1.6	verbose	26
3.22	hstor_keylist Struct Reference	27
3.22.1	Field Documentation	27
3.22.1.1	common_pfx	27
3.22.1.2	contents	27
3.22.1.3	delim	27
3.22.1.4	marker	27
3.22.1.5	max_keys	27
3.22.1.6	name	27
3.22.1.7	prefix	27
3.22.1.8	trunc	27

3.23	hstor_object Struct Reference	28
3.23.1	Field Documentation	28
3.23.1.1	etag	28
3.23.1.2	key	28
3.23.1.3	own_id	28
3.23.1.4	own_name	28
3.23.1.5	size	28
3.23.1.6	storage	28
3.23.1.7	time_mod	28
3.24	http_hdr Struct Reference	29
3.24.1	Field Documentation	29
3.24.1.1	key	29
3.24.1.2	val	29
3.25	http_req Struct Reference	30
3.25.1	Field Documentation	30
3.25.1.1	hdr	30
3.25.1.2	major	30
3.25.1.3	method	30
3.25.1.4	minor	30
3.25.1.5	n_hdr	30
3.25.1.6	orig_path	30
3.25.1.7	uri	30
3.26	http_uri Struct Reference	31
3.26.1	Field Documentation	31
3.26.1.1	fragment	31
3.26.1.2	fragment_len	31
3.26.1.3	hostname	31
3.26.1.4	hostname_len	31
3.26.1.5	path	31
3.26.1.6	path_len	31
3.26.1.7	port	31
3.26.1.8	query	31
3.26.1.9	query_len	31
3.26.1.10	scheme	31
3.26.1.11	scheme_len	31
3.26.1.12	userinfo	31

3.26.1.13	userinfo_len	31
3.27	list_head Struct Reference	32
3.27.1	Field Documentation	32
3.27.1.1	next	32
3.27.1.2	prev	32
3.28	ncld_fh Struct Reference	33
3.28.1	Field Documentation	33
3.28.1.1	errc	33
3.28.1.2	event_arg	33
3.28.1.3	event_func	33
3.28.1.4	event_mask	33
3.28.1.5	fh	33
3.28.1.6	is_open	33
3.28.1.7	nios	33
3.28.1.8	sess	33
3.29	ncld_read Struct Reference	34
3.29.1	Field Documentation	34
3.29.1.1	errc	34
3.29.1.2	fh	34
3.29.1.3	is_done	34
3.29.1.4	length	34
3.29.1.5	meta	34
3.29.1.6	ptr	34
3.30	ncld_sess Struct Reference	35
3.30.1	Field Documentation	36
3.30.1.1	cond	36
3.30.1.2	errc	36
3.30.1.3	event	36
3.30.1.4	event_arg	36
3.30.1.5	handles	36
3.30.1.6	host	36
3.30.1.7	is_up	36
3.30.1.8	mutex	36
3.30.1.9	open_done	36
3.30.1.10	port	36
3.30.1.11	thread	36

3.30.1.12 tlist	36
3.30.1.13 to_thread	36
3.30.1.14 udp	36
3.30.1.15 udp_timer	36
3.31 objcache Struct Reference	37
3.31.1 Field Documentation	37
3.31.1.1 lock	37
3.31.1.2 table	37
3.32 objcache_entry Struct Reference	38
3.32.1 Field Documentation	38
3.32.1.1 flags	38
3.32.1.2 hash	38
3.32.1.3 ref	38
3.33 st_client Struct Reference	39
3.33.1 Field Documentation	39
3.33.1.1 fd	39
3.33.1.2 host	39
3.33.1.3 key	39
3.33.1.4 req_buf	39
3.33.1.5 ssl	39
3.33.1.6 ssl_ctx	39
3.33.1.7 user	39
3.33.1.8 verbose	39
3.34 st_keylist Struct Reference	40
3.34.1 Field Documentation	40
3.34.1.1 contents	40
3.34.1.2 name	40
3.35 st_object Struct Reference	41
3.35.1 Field Documentation	41
3.35.1.1 etag	41
3.35.1.2 name	41
3.35.1.3 owner	41
3.35.1.4 size	41
3.35.1.5 time_mod	41
4 File Documentation	43
4.1 include/chunk-private.h File Reference	43

4.1.1	Define Documentation	43
4.1.1.1	BAD_TPATH_FMT	43
4.1.1.2	MDB_TPATH_FMT	43
4.1.1.3	PREFIX_LEN	43
4.2	include/chunk_msg.h File Reference	44
4.2.1	Define Documentation	44
4.2.1.1	CHUNKD_MAGIC	44
4.2.2	Enumeration Type Documentation	44
4.2.2.1	"@0	44
4.2.2.2	chunk_check_state	45
4.2.2.3	chunk_errcode	45
4.2.2.4	chunk_flags	45
4.2.2.5	chunksrv_ops	45
4.3	include/chunkc.h File Reference	47
4.3.1	Function Documentation	48
4.3.1.1	stc_check_start	48
4.3.1.2	stc_check_status	48
4.3.1.3	stc_cp	48
4.3.1.4	stc_del	48
4.3.1.5	stc_free	48
4.3.1.6	stc_free_keylist	48
4.3.1.7	stc_free_object	48
4.3.1.8	stc_get	48
4.3.1.9	stc_get_inline	48
4.3.1.10	stc_get_recv	48
4.3.1.11	stc_get_start	48
4.3.1.12	stc_init	48
4.3.1.13	stc_keys	48
4.3.1.14	stc_new	48
4.3.1.15	stc_ping	48
4.3.1.16	stc_put	48
4.3.1.17	stc_put_inline	48
4.3.1.18	stc_put_send	48
4.3.1.19	stc_put_start	48
4.3.1.20	stc_put_sync	48
4.3.1.21	stc_readport	48

4.3.1.22	stc_table_open	48
4.4	include/chunksrv.h File Reference	49
4.4.1	Function Documentation	49
4.4.1.1	chreq_sign	49
4.4.1.2	req_len	49
4.5	include/cld-private.h File Reference	50
4.6	include/cld_common.h File Reference	51
4.6.1	Define Documentation	52
4.6.1.1	CLD_ALIGN8	52
4.6.1.2	CLD_PKT_FTR_LEN	52
4.6.1.3	PKT_HDR_TO_STR_SCRATCH_LEN	52
4.6.1.4	SIDARG	52
4.6.1.5	SIDFMT	52
4.6.2	Function Documentation	52
4.6.2.1	__attribute__	52
4.6.2.2	__cld_dump_buf	52
4.6.2.3	cld_authcheck	52
4.6.2.4	cld_authsign	52
4.6.2.5	cld_errstr	52
4.6.2.6	cld_opstr	52
4.6.2.7	cld_pkt_hdr_to_str	52
4.6.2.8	cld_rand64	52
4.6.2.9	cld_readport	52
4.6.2.10	cld_sid2llu	52
4.6.2.11	cld_timer_add	52
4.6.2.12	cld_timer_del	52
4.6.2.13	cld_timers_run	52
4.7	include/cldc.h File Reference	53
4.7.1	Function Documentation	56
4.7.1.1	cldc_close	56
4.7.1.2	cldc_copts_get_data	56
4.7.1.3	cldc_copts_get_metadata	56
4.7.1.4	cldc_del	56
4.7.1.5	cldc_dirent_count	56
4.7.1.6	cldc_dirent_cur_fini	56
4.7.1.7	cldc_dirent_cur_init	56

4.7.1.8	cldc_dirent_first	56
4.7.1.9	cldc_dirent_name	56
4.7.1.10	cldc_dirent_next	56
4.7.1.11	cldc_end_sess	56
4.7.1.12	cldc_get	56
4.7.1.13	cldc_getaddr	56
4.7.1.14	cldc_init	56
4.7.1.15	cldc_kill_sess	56
4.7.1.16	cldc_lock	56
4.7.1.17	cldc_new_sess	56
4.7.1.18	cldc_nop	56
4.7.1.19	cldc_open	56
4.7.1.20	cldc_put	56
4.7.1.21	cldc_receive_pkt	56
4.7.1.22	cldc_saveaddr	57
4.7.1.23	cldc_udp_free	57
4.7.1.24	cldc_udp_new	57
4.7.1.25	cldc_udp_pkt_send	57
4.7.1.26	cldc_udp_receive_pkt	57
4.7.1.27	cldc_unlock	57
4.8	include/elist.h File Reference	58
4.8.1	Define Documentation	58
4.8.1.1	INIT_LIST_HEAD	58
4.8.1.2	list_entry	59
4.8.1.3	list_for_each	59
4.8.1.4	list_for_each_entry	59
4.8.1.5	list_for_each_entry_continue	59
4.8.1.6	list_for_each_entry_safe	59
4.8.1.7	list_for_each_prev	60
4.8.1.8	list_for_each_safe	60
4.8.1.9	LIST_HEAD	60
4.8.1.10	LIST_HEAD_INIT	60
4.9	include/hail_log.h File Reference	61
4.9.1	Define Documentation	61
4.9.1.1	ATTR_PRINTF	61
4.9.1.2	HAIL_CRIT	61

4.9.1.3	HAIL_DEBUG	61
4.9.1.4	HAIL_ERR	62
4.9.1.5	HAIL_INFO	62
4.9.1.6	HAIL_VERBOSE	62
4.9.1.7	HAIL_WARN	62
4.10	include/hail_private.h File Reference	63
4.10.1	Function Documentation	63
4.10.1.1	xdr_sizeof	63
4.11	include/hstor.h File Reference	64
4.11.1	Define Documentation	65
4.11.1.1	ARRAY_SIZE	65
4.11.1.2	PATH_ESCAPE_MASK	65
4.11.1.3	QUERY_ESCAPE_MASK	65
4.11.2	Enumeration Type Documentation	65
4.11.2.1	"@1	65
4.11.2.2	ReqACLC	65
4.11.2.3	ReqQ	66
4.11.3	Function Documentation	68
4.11.3.1	hreq_acl_canned	68
4.11.3.2	hreq_free	68
4.11.3.3	hreq_hdr	68
4.11.3.4	hreq_hdr_push	68
4.11.3.5	hreq_is_query	68
4.11.3.6	hreq_query	68
4.11.3.7	hreq_sign	68
4.11.3.8	hstor_add_bucket	68
4.11.3.9	hstor_del	68
4.11.3.10	hstor_del_bucket	68
4.11.3.11	hstor_free	68
4.11.3.12	hstor_free_blist	68
4.11.3.13	hstor_free_bucket	68
4.11.3.14	hstor_free_keylist	68
4.11.3.15	hstor_free_object	68
4.11.3.16	hstor_get	68
4.11.3.17	hstor_get_inline	68
4.11.3.18	hstor_keys	68

4.11.3.19	hstor_list_buckets	68
4.11.3.20	hstor_new	68
4.11.3.21	hstor_put	68
4.11.3.22	hstor_put_inline	68
4.11.3.23	huri_field_escape	68
4.11.3.24	huri_field_unescape	68
4.11.3.25	huri_parse	68
4.11.3.26	hutil_str2time	68
4.11.3.27	hutil_time2str	68
4.12	include/ncld.h File Reference	69
4.12.1	Function Documentation	70
4.12.1.1	ncld_close	70
4.12.1.2	ncld_del	70
4.12.1.3	ncld_get	70
4.12.1.4	ncld_get_meta	70
4.12.1.5	ncld_init	70
4.12.1.6	ncld_open	70
4.12.1.7	ncld_qlock	70
4.12.1.8	ncld_read_free	70
4.12.1.9	ncld_sess_close	70
4.12.1.10	ncld_sess_open	70
4.12.1.11	ncld_trylock	70
4.12.1.12	ncld_unlock	70
4.12.1.13	ncld_write	70
4.13	include/objcache.h File Reference	71
4.13.1	Define Documentation	71
4.13.1.1	objcache_get	71
4.13.1.2	objcache_get_dirty	71
4.13.1.3	OC_F_DIRTY	71
4.13.2	Function Documentation	71
4.13.2.1	__objcache_get	71
4.13.2.2	objcache_count	71
4.13.2.3	objcache_fini	71
4.13.2.4	objcache_init	71
4.13.2.5	objcache_put	71
4.13.2.6	objcache_test_dirty	71

Chapter 1

Data Structure Index

1.1 Data Structures

Here are the data structures with brief descriptions:

chunk_check_status	5
chunksrv_req	6
chunksrv_resp	7
chunksrv_resp_chkstat	8
chunksrv_resp_get	9
cld_dirent_cur	10
cld_timer	11
cld_timer_list	12
cldc_call_opts (Per-operation application options)	13
cldc_fh (Open file handle associated with a session)	14
cldc_host (Information for a single CLD server host)	15
cldc_msg (Outgoing message, from client to server)	16
cldc_node_metadata	17
cldc_ops (Application-supplied facilities)	18
cldc_pkt_info	19
cldc_session (Single CLD client session)	20
cldc_udp (A UDP implementation of the CLD client protocol)	22
hail_log	23
hstor_blist	24
hstor_bucket	25
hstor_client	26
hstor_keylist	27
hstor_object	28
http_hdr	29
http_req	30
http_uri	31
list_head	32
ncld_fh	33
ncld_read	34
ncld_sess	35
objcache	37
objcache_entry	38
st_client	39

st_keylist	40
st_object	41

Chapter 2

File Index

2.1 File List

Here is a list of all files with brief descriptions:

include/chunk-private.h	43
include/chunk_msg.h	44
include/chunkc.h	47
include/chunksrv.h	49
include/cld-private.h	50
include/cld_common.h	51
include/cldc.h	53
include/elist.h	58
include/hail_log.h	61
include/hail_private.h	63
include/hstor.h	64
include/ncl.h	69
include/objcache.h	71

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 [read]

3.4.1.2 struct [chunksrv_resp](#) chunksrv_resp_chkstat::resp [read]

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` [`read`]

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` [`read`]

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` [`read`]

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` [`read`]

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]` [`read`]

3.12.2.9 `struct cldc_session* cldc_msg::sess` [`read`]

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` `[read]`
- 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` `[read]`
- 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` [`read`]

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] [[read](#)]

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 [[read](#)]

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 [read]

3.27.1.2 struct list_head * list_head::prev [read]

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 [read]

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 [read]

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

- include/[ncld.h](#)

3.29 nclد_read Struct Reference

```
#include <nclد.h>
```

Data Fields

- const void * [ptr](#)
- long [length](#)
- struct [cldc_node_metadata](#) [meta](#)
- struct [nclد_fh](#) * [fh](#)
- bool [is_done](#)
- int [errc](#)

3.29.1 Field Documentation

3.29.1.1 int [nclد_read::errc](#)

3.29.1.2 struct [nclد_fh](#)* [nclد_read::fh](#) [[read](#)]

3.29.1.3 bool [nclد_read::is_done](#)

3.29.1.4 long [nclد_read::length](#)

3.29.1.5 struct [cldc_node_metadata](#) [nclد_read::meta](#) [[read](#)]

3.29.1.6 const void* [nclد_read::ptr](#)

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

- [include/nclد.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` `[read]`
- 3.30.1.13 `int nclد_sess::to_thread[2]`
- 3.30.1.14 `struct cldc_udp* nclد_sess::udp` `[read]`
- 3.30.1.15 `struct cld_timer nclد_sess::udp_timer` `[read]`

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 * pfid, 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 * pfid, 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](#)

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 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](#)

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) { &(amp;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]`
- 4.11.3.26 `time_t hutil_str2time (const char * timestr)`

4.12 include/nclld.h File Reference

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

Data Structures

- struct [nclld_sess](#)
- struct [nclld_fh](#)
- struct [nclld_read](#)

Functions

- struct [nclld_sess](#) * [nclld_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 [nclld_fh](#) * [nclld_open](#) (struct [nclld_sess](#) *s, const char *fname, unsigned int mode, int *error, unsigned int events, void(*event)(void *, unsigned int), void *ev_arg)
- int [nclld_del](#) (struct [nclld_sess](#) *nsess, const char *fname)
- struct [nclld_read](#) * [nclld_get](#) (struct [nclld_fh](#) *fh, int *error)
- struct [nclld_read](#) * [nclld_get_meta](#) (struct [nclld_fh](#) *fh, int *error)
- void [nclld_read_free](#) (struct [nclld_read](#) *rp)
- int [nclld_write](#) (struct [nclld_fh](#) *, const void *data, long len)
- int [nclld_trylock](#) (struct [nclld_fh](#) *)
- int [nclld_qlock](#) (struct [nclld_fh](#) *)
- int [nclld_unlock](#) (struct [nclld_fh](#) *)
- void [nclld_close](#) (struct [nclld_fh](#) *)
- void [nclld_sess_close](#) (struct [nclld_sess](#) *s)
- void [nclld_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, 52](#)
 - [__cld_dump_buf](#)
 - [cld_common.h, 52](#)
 - [__objcache_get](#)
 - [objcache.h, 71](#)
- acc
 - [hstor_client, 26](#)
- ACLC_AUTH_R
 - [hstor.h, 65](#)
- ACLC_PRIV
 - [hstor.h, 65](#)
- ACLC_PUB_R
 - [hstor.h, 65](#)
- ACLC_PUB_RW
 - [hstor.h, 65](#)
- ACLCNUM
 - [hstor.h, 65](#)
- addr
 - [cldc_session, 21](#)
 - [cldc_udp, 22](#)
- addr_len
 - [cldc_session, 21](#)
 - [cldc_udp, 22](#)
- ARRAY_SIZE
 - [hstor.h, 65](#)
- ATTR_PRINTF
 - [hail_log.h, 61](#)
- BAD_TPATH_FMT
 - [chunk-private.h, 43](#)
- cb
 - [cld_timer, 11](#)
 - [cldc_call_opts, 13](#)
 - [cldc_msg, 16](#)
 - [cldc_udp, 22](#)
- cb_private
 - [cldc_msg, 16](#)
 - [cldc_udp, 22](#)
- cfh
 - [cldc_session, 21](#)
- CHD_CSUM_SZ
 - [chunk_msg.h, 44](#)
- CHD_KEY_SZ
 - [chunk_msg.h, 44](#)
- CHD_MAGIC_SZ
 - [chunk_msg.h, 44](#)
- CHD_SIG_SZ
 - [chunk_msg.h, 45](#)
- CHD_USER_SZ
 - [chunk_msg.h, 44](#)
- che_AccessDenied
 - [chunk_msg.h, 45](#)
- che_Busy
 - [chunk_msg.h, 45](#)
- che_InternalError
 - [chunk_msg.h, 45](#)
- che_InvalidArgument
 - [chunk_msg.h, 45](#)
- che_InvalidKey
 - [chunk_msg.h, 45](#)
- che_InvalidTable
 - [chunk_msg.h, 45](#)
- che_InvalidURI
 - [chunk_msg.h, 45](#)
- che_KeyExists
 - [chunk_msg.h, 45](#)
- che_NoSuchKey
 - [chunk_msg.h, 45](#)
- che_SignatureDoesNotMatch
 - [chunk_msg.h, 45](#)
- che_Success
 - [chunk_msg.h, 45](#)
- CHF_SYNC
 - [chunk_msg.h, 45](#)
- CHF_TBL_CREAT
 - [chunk_msg.h, 45](#)
- CHF_TBL_EXCL
 - [chunk_msg.h, 45](#)
- chk_Active
 - [chunk_msg.h, 45](#)
- chk_Idle
 - [chunk_msg.h, 45](#)
- chk_Off
 - [chunk_msg.h, 45](#)
- chkstat
 - [chunksrv_resp_chkstat, 8](#)
- CHO_CHECK_START

- chunk_msg.h, 46
- CHO_CHECK_STATUS
 - chunk_msg.h, 46
- CHO_CP
 - chunk_msg.h, 46
- CHO_DEL
 - chunk_msg.h, 45
- CHO_GET
 - chunk_msg.h, 45
- CHO_GET_META
 - chunk_msg.h, 45
- CHO_LIST
 - chunk_msg.h, 45
- CHO_LOGIN
 - chunk_msg.h, 46
- CHO_NOP
 - chunk_msg.h, 45
- CHO_PUT
 - chunk_msg.h, 45
- CHO_START_TLS
 - chunk_msg.h, 46
- CHO_TABLE_OPEN
 - chunk_msg.h, 46
- chreq_sign
 - chunksrv.h, 49
- chunk-private.h
 - BAD_TPATH_FMT, 43
 - MDB_TPATH_FMT, 43
 - PREFIX_LEN, 43
- chunk_msg.h
 - CHD_CSUM_SZ, 44
 - CHD_KEY_SZ, 44
 - CHD_MAGIC_SZ, 44
 - CHD_SIG_SZ, 45
 - CHD_USER_SZ, 44
 - che_AccessDenied, 45
 - che_Busy, 45
 - che_InternalError, 45
 - che_InvalidArgument, 45
 - che_InvalidKey, 45
 - che_InvalidTable, 45
 - che_InvalidURI, 45
 - che_KeyExists, 45
 - che_NoSuchKey, 45
 - che_SignatureDoesNotMatch, 45
 - che_Success, 45
 - CHF_SYNC, 45
 - CHF_TBL_CREAT, 45
 - CHF_TBL_EXCL, 45
 - chk_Active, 45
 - chk_Idle, 45
 - chk_Off, 45
 - CHO_CHECK_START, 46
 - CHO_CHECK_STATUS, 46
 - CHO_CP, 46
 - CHO_DEL, 45
 - CHO_GET, 45
 - CHO_GET_META, 45
 - CHO_LIST, 45
 - CHO_LOGIN, 46
 - CHO_NOP, 45
 - CHO_PUT, 45
 - CHO_START_TLS, 46
 - CHO_TABLE_OPEN, 46
- chunk_check_state
 - chunk_msg.h, 45
- chunk_check_status, 5
 - count, 5
 - lastdone, 5
 - pad, 5
 - state, 5
- chunk_errcode
 - chunk_msg.h, 45
- chunk_flags
 - chunk_msg.h, 45
- chunk_msg.h
 - chunk_check_state, 45
 - chunk_errcode, 45
 - chunk_flags, 45
 - CHUNKD_MAGIC, 44
 - chunksrv_ops, 45
- chunkc.h
 - stc_check_start, 48
 - stc_check_status, 48
 - stc_cp, 48
 - stc_del, 48
 - stc_free, 48
 - stc_free_keylist, 48
 - stc_free_object, 48
 - stc_get, 48
 - stc_get_inline, 48
 - stc_get_recv, 48
 - stc_get_start, 48
 - stc_init, 48
 - stc_keys, 48
 - stc_new, 48
 - stc_ping, 48
 - stc_put, 48
 - stc_put_inline, 48
 - stc_put_send, 48
 - stc_put_start, 48
 - stc_put_sync, 48
 - stc_readport, 48
 - stc_table_open, 48
- CHUNKD_MAGIC
 - chunk_msg.h, 44
- chunksrv.h
 - chreq_sign, 49

- req_len, [49](#)
- chunksrv_ops
 - chunk_msg.h, [45](#)
- chunksrv_req, [6](#)
 - data_len, [6](#)
 - flags, [6](#)
 - key_len, [6](#)
 - magic, [6](#)
 - nonce, [6](#)
 - op, [6](#)
 - sig, [6](#)
- chunksrv_resp, [7](#)
 - data_len, [7](#)
 - hash, [7](#)
 - magic, [7](#)
 - nonce, [7](#)
 - resp_code, [7](#)
 - rsv1, [7](#)
- chunksrv_resp_chkstat, [8](#)
 - chkstat, [8](#)
 - resp, [8](#)
- chunksrv_resp_get, [9](#)
 - mtime, [9](#)
 - resp, [9](#)
- CLD_ALIGN8
 - cld_common.h, [52](#)
- cld_authcheck
 - cld_common.h, [52](#)
- cld_authsign
 - cld_common.h, [52](#)
- cld_common.h
 - __attribute__, [52](#)
 - __cld_dump_buf, [52](#)
 - CLD_ALIGN8, [52](#)
 - cld_authcheck, [52](#)
 - cld_authsign, [52](#)
 - cld_errstr, [52](#)
 - cld_opstr, [52](#)
 - CLD_PKT_FTR_LEN, [52](#)
 - cld_pkt_hdr_to_str, [52](#)
 - cld_rand64, [52](#)
 - cld_readport, [52](#)
 - cld_sid2llu, [52](#)
 - cld_timer_add, [52](#)
 - cld_timer_del, [52](#)
 - cld_timers_run, [52](#)
 - PKT_HDR_TO_STR_SCRATCH_LEN, [52](#)
 - SIDARG, [52](#)
 - SIDFMT, [52](#)
- cld_dirent_cur, [10](#)
 - p, [10](#)
 - tmp_len, [10](#)
- cld_errstr
 - cld_common.h, [52](#)
- cld_opstr
 - cld_common.h, [52](#)
- CLD_PKT_FTR_LEN
 - cld_common.h, [52](#)
- cld_pkt_hdr_to_str
 - cld_common.h, [52](#)
- cld_rand64
 - cld_common.h, [52](#)
- cld_readport
 - cld_common.h, [52](#)
- cld_sid2llu
 - cld_common.h, [52](#)
- cld_timer, [11](#)
 - cb, [11](#)
 - expires, [11](#)
 - fired, [11](#)
 - name, [11](#)
 - on_list, [11](#)
 - userdata, [11](#)
- cld_timer_add
 - cld_common.h, [52](#)
- cld_timer_del
 - cld_common.h, [52](#)
- cld_timer_list, [12](#)
 - list, [12](#)
 - runmark, [12](#)
- cld_timers_run
 - cld_common.h, [52](#)
- cldc.h
 - cldc_close, [56](#)
 - cldc_copts_get_data, [56](#)
 - cldc_copts_get_metadata, [56](#)
 - cldc_del, [56](#)
 - cldc_dirent_count, [56](#)
 - cldc_dirent_cur_fini, [56](#)
 - cldc_dirent_cur_init, [56](#)
 - cldc_dirent_first, [56](#)
 - cldc_dirent_name, [56](#)
 - cldc_dirent_next, [56](#)
 - cldc_end_sess, [56](#)
 - cldc_get, [56](#)
 - cldc_getaddr, [56](#)
 - cldc_init, [56](#)
 - cldc_kill_sess, [56](#)
 - cldc_lock, [56](#)
 - cldc_new_sess, [56](#)
 - cldc_nop, [56](#)
 - cldc_open, [56](#)
 - cldc_put, [56](#)
 - cldc_receive_pkt, [56](#)
 - cldc_saveaddr, [57](#)
 - cldc_udp_free, [57](#)
 - cldc_udp_new, [57](#)
 - cldc_udp_pkt_send, [57](#)

- cldc_udp_receive_pkt, 57
 - cldc_unlock, 57
- cldc_call_opts, 13
 - cb, 13
 - private, 13
 - resp, 13
- cldc_close
 - cldc.h, 56
- cldc_copts_get_data
 - cldc.h, 56
- cldc_copts_get_metadata
 - cldc.h, 56
- cldc_del
 - cldc.h, 56
- cldc_dirent_count
 - cldc.h, 56
- cldc_dirent_cur_fini
 - cldc.h, 56
- cldc_dirent_cur_init
 - cldc.h, 56
- cldc_dirent_first
 - cldc.h, 56
- cldc_dirent_name
 - cldc.h, 56
- cldc_dirent_next
 - cldc.h, 56
- cldc_end_sess
 - cldc.h, 56
- cldc_fh, 14
 - fh, 14
 - sess, 14
 - valid, 14
- cldc_get
 - cldc.h, 56
- cldc_getaddr
 - cldc.h, 56
- cldc_host, 15
 - host, 15
 - port, 15
 - prio, 15
 - weight, 15
- cldc_init
 - cldc.h, 56
- cldc_kill_sess
 - cldc.h, 56
- cldc_lock
 - cldc.h, 56
- cldc_msg, 16
 - cb, 16
 - cb_private, 16
 - copts, 16
 - done, 16
 - expire_time, 16
 - n_pkts, 16
 - op, 16
 - pkt_info, 16
 - sess, 16
 - xid, 16
- cldc_new_sess
 - cldc.h, 56
- cldc_node_metadata, 17
 - flags, 17
 - inode_name, 17
 - inum, 17
 - time_create, 17
 - time_modify, 17
 - vers, 17
- cldc_nop
 - cldc.h, 56
- cldc_open
 - cldc.h, 56
- cldc_ops, 18
 - event, 18
 - pkt_send, 18
 - timer_ctl, 18
- cldc_pkt_info, 19
 - data, 19
 - hdr_len, 19
 - pkt_len, 19
 - retries, 19
 - user, 19
- cldc_put
 - cldc.h, 56
- cldc_receive_pkt
 - cldc.h, 56
- cldc_saveaddr
 - cldc.h, 57
- cldc_session, 20
 - addr, 21
 - addr_len, 21
 - cfh, 21
 - confirmed, 21
 - expire_time, 21
 - expired, 21
 - inode_name_temp, 21
 - log, 21
 - msg_buf, 21
 - msg_buf_len, 21
 - msg_buf_op, 21
 - msg_scan_time, 21
 - next_seqid_in, 21
 - next_seqid_in_tr, 21
 - next_seqid_out, 21
 - ops, 21
 - out_msg, 21
 - payload, 21
 - private, 21
 - secret_key, 21

- sid, 21
 - user, 21
- cldc_udp, 22
 - addr, 22
 - addr_len, 22
 - cb, 22
 - cb_private, 22
 - fd, 22
 - sess, 22
- cldc_udp_free
 - cldc.h, 57
- cldc_udp_new
 - cldc.h, 57
- cldc_udp_pkt_send
 - cldc.h, 57
- cldc_udp_receive_pkt
 - cldc.h, 57
- cldc_unlock
 - cldc.h, 57
- common_pfx
 - hstor_keylist, 27
- cond
 - ncld_sess, 36
- confirmed
 - cldc_session, 21
- contents
 - hstor_keylist, 27
 - st_keylist, 40
- copts
 - cldc_msg, 16
- count
 - chunk_check_status, 5
- curl
 - hstor_client, 26
- data
 - cldc_pkt_info, 19
- data_len
 - chunksrv_req, 6
 - chunksrv_resp, 7
- debug
 - hail_log, 23
- delim
 - hstor_keylist, 27
- done
 - cldc_msg, 16
- elist.h
 - INIT_LIST_HEAD, 58
 - list_entry, 58
 - list_for_each, 59
 - list_for_each_entry, 59
 - list_for_each_entry_continue, 59
 - list_for_each_entry_safe, 59
 - list_for_each_prev, 59
 - list_for_each_safe, 60
 - LIST_HEAD, 60
 - LIST_HEAD_INIT, 60
- errc
 - ncld_fh, 33
 - ncld_read, 34
 - ncld_sess, 36
- etag
 - hstor_object, 28
 - st_object, 41
- event
 - cldc_ops, 18
 - ncld_sess, 36
- event_arg
 - ncld_fh, 33
 - ncld_sess, 36
- event_func
 - ncld_fh, 33
- event_mask
 - ncld_fh, 33
- expire_time
 - cldc_msg, 16
 - cldc_session, 21
- expired
 - cldc_session, 21
- expires
 - cld_timer, 11
- fd
 - cldc_udp, 22
 - st_client, 39
- fh
 - cldc_fh, 14
 - ncld_fh, 33
 - ncld_read, 34
- fired
 - cld_timer, 11
- flags
 - chunksrv_req, 6
 - cldc_node_metadata, 17
 - objcache_entry, 38
- fragment
 - http_uri, 31
- fragment_len
 - http_uri, 31
- func
 - hail_log, 23
- HAIL_CRIT
 - hail_log.h, 61
- HAIL_DEBUG
 - hail_log.h, 61
- HAIL_ERR

- hail_log.h, 61
- HAIL_INFO
 - hail_log.h, 62
- hail_log, 23
 - debug, 23
 - func, 23
 - verbose, 23
- hail_log.h
 - ATTR_PRINTF, 61
 - HAIL_CRIT, 61
 - HAIL_DEBUG, 61
 - HAIL_ERR, 61
 - HAIL_INFO, 62
 - HAIL_VERBOSE, 62
 - HAIL_WARN, 62
- hail_private.h
 - xdr_sizeof, 63
- HAIL_VERBOSE
 - hail_log.h, 62
- HAIL_WARN
 - hail_log.h, 62
- handles
 - ncld_sess, 36
- hash
 - chunksrv_resp, 7
 - objcache_entry, 38
- hdr
 - http_req, 30
- hdr_len
 - cldc_pkt_info, 19
- host
 - cldc_host, 15
 - hstor_client, 26
 - ncld_sess, 36
 - st_client, 39
- hostname
 - http_uri, 31
- hostname_len
 - http_uri, 31
- HREQ_MAX_HDR
 - hstor.h, 65
- hreq_acl_canned
 - hstor.h, 68
- hreq_free
 - hstor.h, 68
- hreq_hdr
 - hstor.h, 68
- hreq_hdr_push
 - hstor.h, 68
- hreq_is_query
 - hstor.h, 68
- hreq_query
 - hstor.h, 68
- hreq_sign
 - hstor.h, 68
- hstor.h, 68
- hstor.h
 - ACLC_AUTH_R, 65
 - ACLC_PRIV, 65
 - ACLC_PUB_R, 65
 - ACLC_PUB_RW, 65
 - ACLCNUM, 65
 - ARRAY_SIZE, 65
 - HREQ_MAX_HDR, 65
 - hreq_acl_canned, 68
 - hreq_free, 68
 - hreq_hdr, 68
 - hreq_hdr_push, 68
 - hreq_is_query, 68
 - hreq_query, 68
 - hreq_sign, 68
 - hstor_add_bucket, 68
 - hstor_del, 68
 - hstor_del_bucket, 68
 - hstor_free, 68
 - hstor_free_blist, 68
 - hstor_free_bucket, 68
 - hstor_free_keylist, 68
 - hstor_free_object, 68
 - hstor_get, 68
 - hstor_get_inline, 68
 - hstor_keys, 68
 - hstor_list_buckets, 68
 - hstor_new, 68
 - hstor_put, 68
 - hstor_put_inline, 68
 - huri_field_escape, 68
 - huri_field_unescape, 68
 - huri_parse, 68
 - hutil_str2time, 68
 - hutil_time2str, 68
 - PATH_ESCAPE_MASK, 65
 - QUERY_ESCAPE_MASK, 65
 - ReqACLC, 65
 - ReqQ, 65
 - URIQ_ACL, 66
 - URIQ_LOCATION, 66
 - URIQ_LOGGING, 66
 - URIQ_TORRENT, 66
 - URIQNUM, 66
- hstor_add_bucket
 - hstor.h, 68
- hstor_blist, 24
 - list, 24
 - own_id, 24
 - own_name, 24
- hstor_bucket, 25
 - name, 25
 - time_create, 25

- hstor_client, 26
 - acc, 26
 - curl, 26
 - host, 26
 - key, 26
 - user, 26
 - verbose, 26
- hstor_del
 - hstor.h, 68
- hstor_del_bucket
 - hstor.h, 68
- hstor_free
 - hstor.h, 68
- hstor_free_blist
 - hstor.h, 68
- hstor_free_bucket
 - hstor.h, 68
- hstor_free_keylist
 - hstor.h, 68
- hstor_free_object
 - hstor.h, 68
- hstor_get
 - hstor.h, 68
- hstor_get_inline
 - hstor.h, 68
- hstor_keylist, 27
 - common_pfx, 27
 - contents, 27
 - delim, 27
 - marker, 27
 - max_keys, 27
 - name, 27
 - prefix, 27
 - trunc, 27
- hstor_keys
 - hstor.h, 68
- hstor_list_buckets
 - hstor.h, 68
- hstor_new
 - hstor.h, 68
- hstor_object, 28
 - etag, 28
 - key, 28
 - own_id, 28
 - own_name, 28
 - size, 28
 - storage, 28
 - time_mod, 28
- hstor_put
 - hstor.h, 68
- hstor_put_inline
 - hstor.h, 68
- http_hdr, 29
 - key, 29
 - val, 29
- http_req, 30
 - hdr, 30
 - major, 30
 - method, 30
 - minor, 30
 - n_hdr, 30
 - orig_path, 30
 - uri, 30
- http_uri, 31
 - fragment, 31
 - fragment_len, 31
 - hostname, 31
 - hostname_len, 31
 - path, 31
 - path_len, 31
 - port, 31
 - query, 31
 - query_len, 31
 - scheme, 31
 - scheme_len, 31
 - userinfo, 31
 - userinfo_len, 31
- huri_field_escape
 - hstor.h, 68
- huri_field_unescape
 - hstor.h, 68
- huri_parse
 - hstor.h, 68
- hutil_str2time
 - hstor.h, 68
- hutil_time2str
 - hstor.h, 68
- include/chunk-private.h, 43
- include/chunk_msg.h, 44
- include/chunkc.h, 47
- include/chunksrv.h, 49
- include/cld-private.h, 50
- include/cld_common.h, 51
- include/cldc.h, 53
- include/elist.h, 58
- include/hail_log.h, 61
- include/hail_private.h, 63
- include/hstor.h, 64
- include/nclld.h, 69
- include/objcache.h, 71
- INIT_LIST_HEAD
 - elist.h, 58
- inode_name
 - cldc_node_metadata, 17
- inode_name_temp
 - cldc_session, 21
- inum

- cldc_node_metadata, 17
- is_done
 - ncld_read, 34
- is_open
 - ncld_fh, 33
- is_up
 - ncld_sess, 36
- key
 - hstor_client, 26
 - hstor_object, 28
 - http_hdr, 29
 - st_client, 39
- key_len
 - chunksrv_req, 6
- lastdone
 - chunk_check_status, 5
- length
 - ncld_read, 34
- list
 - cld_timer_list, 12
 - hstor_blist, 24
- list_entry
 - elist.h, 58
- list_for_each
 - elist.h, 59
- list_for_each_entry
 - elist.h, 59
- list_for_each_entry_continue
 - elist.h, 59
- list_for_each_entry_safe
 - elist.h, 59
- list_for_each_prev
 - elist.h, 59
- list_for_each_safe
 - elist.h, 60
- LIST_HEAD
 - elist.h, 60
- list_head, 32
 - next, 32
 - prev, 32
- LIST_HEAD_INIT
 - elist.h, 60
- lock
 - objcache, 37
- log
 - cldc_session, 21
- magic
 - chunksrv_req, 6
 - chunksrv_resp, 7
- major
 - http_req, 30
- marker
 - hstor_keylist, 27
- max_keys
 - hstor_keylist, 27
- MDB_TPATH_FMT
 - chunk-private.h, 43
- meta
 - ncld_read, 34
- method
 - http_req, 30
- minor
 - http_req, 30
- msg_buf
 - cldc_session, 21
- msg_buf_len
 - cldc_session, 21
- msg_buf_op
 - cldc_session, 21
- msg_scan_time
 - cldc_session, 21
- mtime
 - chunksrv_resp_get, 9
- mutex
 - ncld_sess, 36
- n_hdr
 - http_req, 30
- n_pkts
 - cldc_msg, 16
- name
 - cld_timer, 11
 - hstor_bucket, 25
 - hstor_keylist, 27
 - st_keylist, 40
 - st_object, 41
- ncld.h
 - ncld_close, 70
 - ncld_del, 70
 - ncld_get, 70
 - ncld_get_meta, 70
 - ncld_init, 70
 - ncld_open, 70
 - ncld_qlock, 70
 - ncld_read_free, 70
 - ncld_sess_close, 70
 - ncld_sess_open, 70
 - ncld_trylock, 70
 - ncld_unlock, 70
 - ncld_write, 70
- ncld_close
 - ncld.h, 70
- ncld_del
 - ncld.h, 70
- ncld_fh, 33

- errc, 33
- event_arg, 33
- event_func, 33
- event_mask, 33
- fh, 33
- is_open, 33
- nios, 33
- sess, 33
- ncld_get
 - ncld.h, 70
- ncld_get_meta
 - ncld.h, 70
- ncld_init
 - ncld.h, 70
- ncld_open
 - ncld.h, 70
- ncld_qlock
 - ncld.h, 70
- ncld_read, 34
 - errc, 34
 - fh, 34
 - is_done, 34
 - length, 34
 - meta, 34
 - ptr, 34
- ncld_read_free
 - ncld.h, 70
- ncld_sess, 35
 - cond, 36
 - errc, 36
 - event, 36
 - event_arg, 36
 - handles, 36
 - host, 36
 - is_up, 36
 - mutex, 36
 - open_done, 36
 - port, 36
 - thread, 36
 - tlist, 36
 - to_thread, 36
 - udp, 36
 - udp_timer, 36
- ncld_sess_close
 - ncld.h, 70
- ncld_sess_open
 - ncld.h, 70
- ncld_trylock
 - ncld.h, 70
- ncld_unlock
 - ncld.h, 70
- ncld_write
 - ncld.h, 70
- next
 - list_head, 32
- next_seqid_in
 - cldc_session, 21
- next_seqid_in_tr
 - cldc_session, 21
- next_seqid_out
 - cldc_session, 21
- nios
 - ncld_fh, 33
- nonce
 - chunksrv_req, 6
 - chunksrv_resp, 7
- objcache, 37
 - lock, 37
 - table, 37
- objcache.h
 - __objcache_get, 71
 - objcache_count, 71
 - objcache_fini, 71
 - objcache_get, 71
 - objcache_get_dirty, 71
 - objcache_init, 71
 - objcache_put, 71
 - objcache_test_dirty, 71
 - OC_F_DIRTY, 71
- objcache_count
 - objcache.h, 71
- objcache_entry, 38
 - flags, 38
 - hash, 38
 - ref, 38
- objcache_fini
 - objcache.h, 71
- objcache_get
 - objcache.h, 71
- objcache_get_dirty
 - objcache.h, 71
- objcache_init
 - objcache.h, 71
- objcache_put
 - objcache.h, 71
- objcache_test_dirty
 - objcache.h, 71
- OC_F_DIRTY
 - objcache.h, 71
- on_list
 - cld_timer, 11
- op
 - chunksrv_req, 6
 - cldc_msg, 16
- open_done
 - ncld_sess, 36
- ops

- cldc_session, 21
- orig_path
 - http_req, 30
- out_msg
 - cldc_session, 21
- own_id
 - hstor_blist, 24
 - hstor_object, 28
- own_name
 - hstor_blist, 24
 - hstor_object, 28
- owner
 - st_object, 41
- p
 - cld_dirent_cur, 10
- pad
 - chunk_check_status, 5
- path
 - http_uri, 31
- PATH_ESCAPE_MASK
 - hstor.h, 65
- path_len
 - http_uri, 31
- payload
 - cldc_session, 21
- PKT_HDR_TO_STR_SCRATCH_LEN
 - cld_common.h, 52
- pkt_info
 - cldc_msg, 16
- pkt_len
 - cldc_pkt_info, 19
- pkt_send
 - cldc_ops, 18
- port
 - cldc_host, 15
 - http_uri, 31
 - ncld_sess, 36
- prefix
 - hstor_keylist, 27
- PREFIX_LEN
 - chunk-private.h, 43
- prev
 - list_head, 32
- prio
 - cldc_host, 15
- private
 - cldc_call_opts, 13
 - cldc_session, 21
- ptr
 - ncld_read, 34
- query
 - http_uri, 31
- QUERY_ESCAPE_MASK
 - hstor.h, 65
- query_len
 - http_uri, 31
- ref
 - objcache_entry, 38
- req_buf
 - st_client, 39
- req_len
 - chunksrv.h, 49
- ReqACLC
 - hstor.h, 65
- ReqQ
 - hstor.h, 65
- resp
 - chunksrv_resp_chkstat, 8
 - chunksrv_resp_get, 9
 - cldc_call_opts, 13
- resp_code
 - chunksrv_resp, 7
- retries
 - cldc_pkt_info, 19
- rsvl
 - chunksrv_resp, 7
- runmark
 - cld_timer_list, 12
- scheme
 - http_uri, 31
- scheme_len
 - http_uri, 31
- secret_key
 - cldc_session, 21
- sess
 - cldc_fh, 14
 - cldc_msg, 16
 - cldc_udp, 22
 - ncld_fh, 33
- sid
 - cldc_session, 21
- SIDARG
 - cld_common.h, 52
- SIDFMT
 - cld_common.h, 52
- sig
 - chunksrv_req, 6
- size
 - hstor_object, 28
 - st_object, 41
- ssl
 - st_client, 39
- ssl_ctx
 - st_client, 39

- st_client, 39
 - fd, 39
 - host, 39
 - key, 39
 - req_buf, 39
 - ssl, 39
 - ssl_ctx, 39
 - user, 39
 - verbose, 39
- st_keylist, 40
 - contents, 40
 - name, 40
- st_object, 41
 - etag, 41
 - name, 41
 - owner, 41
 - size, 41
 - time_mod, 41
- state
 - chunk_check_status, 5
- stc_check_start
 - chunkc.h, 48
- stc_check_status
 - chunkc.h, 48
- stc_cp
 - chunkc.h, 48
- stc_del
 - chunkc.h, 48
- stc_free
 - chunkc.h, 48
- stc_free_keylist
 - chunkc.h, 48
- stc_free_object
 - chunkc.h, 48
- stc_get
 - chunkc.h, 48
- stc_get_inline
 - chunkc.h, 48
- stc_get_recv
 - chunkc.h, 48
- stc_get_start
 - chunkc.h, 48
- stc_init
 - chunkc.h, 48
- stc_keys
 - chunkc.h, 48
- stc_new
 - chunkc.h, 48
- stc_ping
 - chunkc.h, 48
- stc_put
 - chunkc.h, 48
- stc_put_inline
 - chunkc.h, 48
- stc_put_send
 - chunkc.h, 48
- stc_put_start
 - chunkc.h, 48
- stc_put_sync
 - chunkc.h, 48
- stc_readport
 - chunkc.h, 48
- stc_table_open
 - chunkc.h, 48
- storage
 - hstor_object, 28
- table
 - objcache, 37
- thread
 - ncld_sess, 36
- time_create
 - cldc_node_metadata, 17
 - hstor_bucket, 25
- time_mod
 - hstor_object, 28
 - st_object, 41
- time_modify
 - cldc_node_metadata, 17
- timer_ctl
 - cldc_ops, 18
- tlist
 - ncld_sess, 36
- tmp_len
 - cld_dirent_cur, 10
- to_thread
 - ncld_sess, 36
- trunc
 - hstor_keylist, 27
- udp
 - ncld_sess, 36
- udp_timer
 - ncld_sess, 36
- uri
 - http_req, 30
- URIQ_ACL
 - hstor.h, 66
- URIQ_LOCATION
 - hstor.h, 66
- URIQ_LOGGING
 - hstor.h, 66
- URIQ_TORRENT
 - hstor.h, 66
- URIQNUM
 - hstor.h, 66
- user
 - cldc_pkt_info, 19

- cldc_session, [21](#)
 - hstor_client, [26](#)
 - st_client, [39](#)
- userdata
 - cld_timer, [11](#)
- userinfo
 - http_uri, [31](#)
- userinfo_len
 - http_uri, [31](#)
- val
 - http_hdr, [29](#)
- valid
 - cldc_fh, [14](#)
- verbose
 - hail_log, [23](#)
 - hstor_client, [26](#)
 - st_client, [39](#)
- vers
 - cldc_node_metadata, [17](#)
- weight
 - cldc_host, [15](#)
- xdr_sizeof
 - hail_private.h, [63](#)
- xid
 - cldc_msg, [16](#)