

CLD

0.1git

Generated by Doxygen 1.8.7

Tue Nov 25 2014 00:27:58



# Contents

<b>1</b>	<b>Data Structure Index</b>	<b>1</b>
1.1	Data Structures . . . . .	1
<b>2</b>	<b>File Index</b>	<b>3</b>
2.1	File List . . . . .	3
<b>3</b>	<b>Data Structure Documentation</b>	<b>5</b>
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 . . . . .	6
3.3.1.1	data_len . . . . .	6
3.3.1.2	hash . . . . .	6
3.3.1.3	magic . . . . .	6
3.3.1.4	nonce . . . . .	6
3.3.1.5	resp_code . . . . .	6
3.3.1.6	rsv1 . . . . .	6
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	7
3.5.1.1	mtime	7
3.5.1.2	resp	7
3.6	cld_dirent_cur Struct Reference	7
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	8
3.7.1.1	cb	8
3.7.1.2	expires	8
3.7.1.3	fired	8
3.7.1.4	name	8
3.7.1.5	on_list	8
3.7.1.6	userdata	8
3.8	cld_timer_list Struct Reference	8
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	9
3.9.2	Field Documentation	9
3.9.2.1	cb	9
3.9.2.2	private	9
3.9.2.3	resp	9
3.10	cldc_fh Struct Reference	9
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	10
3.11.2	Field Documentation	10
3.11.2.1	host	10
3.11.2.2	port	10
3.11.2.3	prio	10

3.11.2.4	weight	10
3.12	cldc_msg Struct Reference	10
3.12.1	Detailed Description	11
3.12.2	Field Documentation	11
3.12.2.1	cb	11
3.12.2.2	cb_private	11
3.12.2.3	copts	11
3.12.2.4	done	11
3.12.2.5	expire_time	11
3.12.2.6	n_pkts	11
3.12.2.7	op	11
3.12.2.8	pkt_info	11
3.12.2.9	sess	11
3.12.2.10	xid	11
3.13	cldc_node_metadata Struct Reference	11
3.13.1	Field Documentation	12
3.13.1.1	flags	12
3.13.1.2	inode_name	12
3.13.1.3	inum	12
3.13.1.4	time_create	12
3.13.1.5	time_modify	12
3.13.1.6	vers	12
3.14	cldc_ops Struct Reference	12
3.14.1	Detailed Description	12
3.14.2	Field Documentation	12
3.14.2.1	event	12
3.14.2.2	pkt_send	12
3.14.2.3	timer_ctl	12
3.15	cldc_pkt_info Struct Reference	13
3.15.1	Field Documentation	13
3.15.1.1	data	13
3.15.1.2	hdr_len	13
3.15.1.3	pkt_len	13
3.15.1.4	retries	13
3.15.1.5	user	13
3.16	cldc_session Struct Reference	13
3.16.1	Detailed Description	14
3.16.2	Field Documentation	14
3.16.2.1	addr	14
3.16.2.2	addr_len	14

3.16.2.3	cfh	14
3.16.2.4	confirmed	14
3.16.2.5	expire_time	14
3.16.2.6	expired	14
3.16.2.7	inode_name_temp	14
3.16.2.8	log	14
3.16.2.9	msg_buf	14
3.16.2.10	msg_buf_len	14
3.16.2.11	msg_buf_op	14
3.16.2.12	msg_scan_time	14
3.16.2.13	next_seqid_in	14
3.16.2.14	next_seqid_in_tr	14
3.16.2.15	next_seqid_out	14
3.16.2.16	ops	14
3.16.2.17	out_msg	14
3.16.2.18	payload	14
3.16.2.19	private	14
3.16.2.20	secret_key	14
3.16.2.21	sid	14
3.16.2.22	user	15
3.17	cldc_udp Struct Reference	15
3.17.1	Detailed Description	15
3.17.2	Field Documentation	15
3.17.2.1	addr	15
3.17.2.2	addr_len	15
3.17.2.3	cb	15
3.17.2.4	cb_private	15
3.17.2.5	fd	15
3.17.2.6	sess	15
3.18	hail_log Struct Reference	15
3.18.1	Field Documentation	16
3.18.1.1	debug	16
3.18.1.2	func	16
3.18.1.3	verbose	16
3.19	hstor_blist Struct Reference	16
3.19.1	Field Documentation	16
3.19.1.1	list	16
3.19.1.2	own_id	16
3.19.1.3	own_name	16
3.20	hstor_bucket Struct Reference	16

3.20.1	Field Documentation	16
3.20.1.1	name	16
3.20.1.2	time_create	16
3.21	hstor_client Struct Reference	17
3.21.1	Field Documentation	17
3.21.1.1	acc	17
3.21.1.2	curl	17
3.21.1.3	host	17
3.21.1.4	key	17
3.21.1.5	subdomain	17
3.21.1.6	user	17
3.21.1.7	verbose	17
3.22	hstor_keylist Struct Reference	17
3.22.1	Field Documentation	18
3.22.1.1	common_pfx	18
3.22.1.2	contents	18
3.22.1.3	delim	18
3.22.1.4	marker	18
3.22.1.5	max_keys	18
3.22.1.6	name	18
3.22.1.7	prefix	18
3.22.1.8	trunc	18
3.23	hstor_object Struct Reference	18
3.23.1	Field Documentation	18
3.23.1.1	etag	18
3.23.1.2	key	18
3.23.1.3	own_id	18
3.23.1.4	own_name	18
3.23.1.5	size	18
3.23.1.6	storage	18
3.23.1.7	time_mod	18
3.24	http_hdr Struct Reference	19
3.24.1	Field Documentation	19
3.24.1.1	key	19
3.24.1.2	val	19
3.25	http_req Struct Reference	19
3.25.1	Field Documentation	19
3.25.1.1	hdr	19
3.25.1.2	major	19
3.25.1.3	method	19

3.25.1.4	minor	19
3.25.1.5	n_hdr	19
3.25.1.6	orig_path	19
3.25.1.7	uri	20
3.26	http_uri Struct Reference	20
3.26.1	Field Documentation	20
3.26.1.1	fragment	20
3.26.1.2	fragment_len	20
3.26.1.3	hostname	20
3.26.1.4	hostname_len	20
3.26.1.5	path	20
3.26.1.6	path_len	20
3.26.1.7	port	20
3.26.1.8	query	20
3.26.1.9	query_len	20
3.26.1.10	scheme	20
3.26.1.11	scheme_len	20
3.26.1.12	userinfo	20
3.26.1.13	userinfo_len	21
3.27	list_head Struct Reference	21
3.27.1	Field Documentation	21
3.27.1.1	next	21
3.27.1.2	prev	21
3.28	ncld_fh Struct Reference	21
3.28.1	Field Documentation	21
3.28.1.1	errc	21
3.28.1.2	event_arg	21
3.28.1.3	event_func	21
3.28.1.4	event_mask	21
3.28.1.5	fh	22
3.28.1.6	is_open	22
3.28.1.7	nios	22
3.28.1.8	sess	22
3.29	ncld_read Struct Reference	22
3.29.1	Field Documentation	22
3.29.1.1	errc	22
3.29.1.2	fh	22
3.29.1.3	is_done	22
3.29.1.4	length	22
3.29.1.5	meta	22



3.29.1.6 ptr . . . . .	22
3.30 nclد_sess Struct Reference . . . . .	22
3.30.1 Field Documentation . . . . .	23
3.30.1.1 cond . . . . .	23
3.30.1.2 errc . . . . .	23
3.30.1.3 event . . . . .	23
3.30.1.4 event_arg . . . . .	23
3.30.1.5 handles . . . . .	23
3.30.1.6 host . . . . .	23
3.30.1.7 is_up . . . . .	23
3.30.1.8 mutex . . . . .	23
3.30.1.9 open_done . . . . .	23
3.30.1.10 port . . . . .	23
3.30.1.11 thread . . . . .	23
3.30.1.12 tlist . . . . .	23
3.30.1.13 to_thread . . . . .	23
3.30.1.14 udp . . . . .	23
3.30.1.15 udp_timer . . . . .	23
3.31 objcache Struct Reference . . . . .	23
3.31.1 Field Documentation . . . . .	24
3.31.1.1 lock . . . . .	24
3.31.1.2 table . . . . .	24
3.32 objcache_entry Struct Reference . . . . .	24
3.32.1 Field Documentation . . . . .	24
3.32.1.1 flags . . . . .	24
3.32.1.2 hash . . . . .	24
3.32.1.3 ref . . . . .	24
3.33 st_client Struct Reference . . . . .	24
3.33.1 Field Documentation . . . . .	25
3.33.1.1 fd . . . . .	25
3.33.1.2 host . . . . .	25
3.33.1.3 key . . . . .	25
3.33.1.4 req_buf . . . . .	25
3.33.1.5 ssl . . . . .	25
3.33.1.6 ssl_ctx . . . . .	25
3.33.1.7 user . . . . .	25
3.33.1.8 verbose . . . . .	25
3.34 st_keylist Struct Reference . . . . .	25
3.34.1 Field Documentation . . . . .	25
3.34.1.1 contents . . . . .	25

3.34.1.2	name	25
3.35	st_object Struct Reference	25
3.35.1	Field Documentation	26
3.35.1.1	etag	26
3.35.1.2	name	26
3.35.1.3	owner	26
3.35.1.4	size	26
3.35.1.5	time_mod	26
<b>4</b>	<b>File Documentation</b>	<b>27</b>
4.1	include/chunk-private.h File Reference	27
4.1.1	Macro Definition Documentation	27
4.1.1.1	BAD_TPATH_FMT	27
4.1.1.2	MDB_TPATH_FMT	27
4.1.1.3	PREFIX_LEN	27
4.2	include/chunk_msg.h File Reference	27
4.2.1	Macro Definition Documentation	28
4.2.1.1	CHUNKD_MAGIC	28
4.2.2	Enumeration Type Documentation	28
4.2.2.1	anonymous enum	28
4.2.2.2	chunk_check_state	28
4.2.2.3	chunk_errcode	28
4.2.2.4	chunk_flags	29
4.2.2.5	chunksrv_ops	29
4.3	include/chunkc.h File Reference	29
4.3.1	Function Documentation	30
4.3.1.1	stc_check_start	30
4.3.1.2	stc_check_status	30
4.3.1.3	stc_cp	30
4.3.1.4	stc_del	30
4.3.1.5	stc_free	30
4.3.1.6	stc_free_keylist	30
4.3.1.7	stc_free_object	30
4.3.1.8	stc_get	30
4.3.1.9	stc_get_inline	30
4.3.1.10	stc_get_recv	30
4.3.1.11	stc_get_start	31
4.3.1.12	stc_init	31
4.3.1.13	stc_keys	31
4.3.1.14	stc_new	31

4.3.1.15	<a href="#">stc_ping</a>	31
4.3.1.16	<a href="#">stc_put</a>	31
4.3.1.17	<a href="#">stc_put_inline</a>	31
4.3.1.18	<a href="#">stc_put_send</a>	31
4.3.1.19	<a href="#">stc_put_start</a>	31
4.3.1.20	<a href="#">stc_put_sync</a>	31
4.3.1.21	<a href="#">stc_readport</a>	31
4.3.1.22	<a href="#">stc_table_open</a>	31
4.4	<a href="#">include/chunksrv.h File Reference</a>	31
4.4.1	<a href="#">Function Documentation</a>	31
4.4.1.1	<a href="#">chreq_sign</a>	31
4.4.1.2	<a href="#">req_len</a>	31
4.5	<a href="#">include/cld-private.h File Reference</a>	31
4.6	<a href="#">include/cld_common.h File Reference</a>	32
4.6.1	<a href="#">Macro Definition Documentation</a>	32
4.6.1.1	<a href="#">CLD_ALIGN8</a>	32
4.6.1.2	<a href="#">CLD_PKT_FTR_LEN</a>	32
4.6.1.3	<a href="#">PKT_HDR_TO_STR_SCRATCH_LEN</a>	33
4.6.1.4	<a href="#">SIDARG</a>	33
4.6.1.5	<a href="#">SIDFMT</a>	33
4.6.2	<a href="#">Function Documentation</a>	33
4.6.2.1	<a href="#">__attribute__</a>	33
4.6.2.2	<a href="#">__cld_dump_buf</a>	33
4.6.2.3	<a href="#">cld_authcheck</a>	33
4.6.2.4	<a href="#">cld_authsign</a>	33
4.6.2.5	<a href="#">cld_errstr</a>	33
4.6.2.6	<a href="#">cld_opstr</a>	33
4.6.2.7	<a href="#">cld_pkt_hdr_to_str</a>	33
4.6.2.8	<a href="#">cld_rand64</a>	33
4.6.2.9	<a href="#">cld_readport</a>	33
4.6.2.10	<a href="#">cld_sid2llu</a>	33
4.6.2.11	<a href="#">cld_timer_add</a>	33
4.6.2.12	<a href="#">cld_timer_del</a>	33
4.6.2.13	<a href="#">cld_timers_run</a>	33
4.7	<a href="#">include/cldc.h File Reference</a>	33
4.7.1	<a href="#">Function Documentation</a>	35
4.7.1.1	<a href="#">cldc_close</a>	35
4.7.1.2	<a href="#">cldc_copts_get_data</a>	35
4.7.1.3	<a href="#">cldc_copts_get_metadata</a>	35
4.7.1.4	<a href="#">cldc_del</a>	35

4.7.1.5	<a href="#">cldc_dirent_count</a>	35
4.7.1.6	<a href="#">cldc_dirent_cur_fini</a>	35
4.7.1.7	<a href="#">cldc_dirent_cur_init</a>	35
4.7.1.8	<a href="#">cldc_dirent_first</a>	35
4.7.1.9	<a href="#">cldc_dirent_name</a>	35
4.7.1.10	<a href="#">cldc_dirent_next</a>	35
4.7.1.11	<a href="#">cldc_end_sess</a>	35
4.7.1.12	<a href="#">cldc_get</a>	35
4.7.1.13	<a href="#">cldc_getaddr</a>	35
4.7.1.14	<a href="#">cldc_init</a>	35
4.7.1.15	<a href="#">cldc_kill_sess</a>	35
4.7.1.16	<a href="#">cldc_lock</a>	35
4.7.1.17	<a href="#">cldc_new_sess</a>	35
4.7.1.18	<a href="#">cldc_nop</a>	35
4.7.1.19	<a href="#">cldc_open</a>	35
4.7.1.20	<a href="#">cldc_put</a>	35
4.7.1.21	<a href="#">cldc_receive_pkt</a>	35
4.7.1.22	<a href="#">cldc_saveaddr</a>	36
4.7.1.23	<a href="#">cldc_udp_free</a>	36
4.7.1.24	<a href="#">cldc_udp_new</a>	36
4.7.1.25	<a href="#">cldc_udp_pkt_send</a>	36
4.7.1.26	<a href="#">cldc_udp_receive_pkt</a>	36
4.7.1.27	<a href="#">cldc_unlock</a>	36
4.8	<a href="#">include/elist.h File Reference</a>	36
4.8.1	<a href="#">Macro Definition Documentation</a>	37
4.8.1.1	<a href="#">INIT_LIST_HEAD</a>	37
4.8.1.2	<a href="#">list_entry</a>	37
4.8.1.3	<a href="#">list_for_each</a>	37
4.8.1.4	<a href="#">list_for_each_entry</a>	37
4.8.1.5	<a href="#">list_for_each_entry_continue</a>	37
4.8.1.6	<a href="#">list_for_each_entry_safe</a>	37
4.8.1.7	<a href="#">list_for_each_prev</a>	38
4.8.1.8	<a href="#">list_for_each_safe</a>	38
4.8.1.9	<a href="#">LIST_HEAD</a>	38
4.8.1.10	<a href="#">LIST_HEAD_INIT</a>	38
4.9	<a href="#">include/hail_log.h File Reference</a>	38
4.9.1	<a href="#">Macro Definition Documentation</a>	39
4.9.1.1	<a href="#">ATTR_PRINTF</a>	39
4.9.1.2	<a href="#">HAIL_CRIT</a>	39
4.9.1.3	<a href="#">HAIL_DEBUG</a>	39

4.9.1.4	HAIL_ERR . . . . .	39
4.9.1.5	HAIL_INFO . . . . .	39
4.9.1.6	HAIL_VERBOSE . . . . .	39
4.9.1.7	HAIL_WARN . . . . .	39
4.10	include/hail_private.h File Reference . . . . .	39
4.11	include/hstor.h File Reference . . . . .	40
4.11.1	Macro Definition Documentation . . . . .	41
4.11.1.1	ARRAY_SIZE . . . . .	41
4.11.1.2	PATH_ESCAPE_MASK . . . . .	41
4.11.1.3	QUERY_ESCAPE_MASK . . . . .	41
4.11.2	Enumeration Type Documentation . . . . .	41
4.11.2.1	anonymous enum . . . . .	41
4.11.2.2	hstor_calling_format . . . . .	41
4.11.2.3	ReqACLC . . . . .	41
4.11.2.4	ReqQ . . . . .	42
4.11.3	Function Documentation . . . . .	42
4.11.3.1	hreq_acl_canned . . . . .	42
4.11.3.2	hreq_free . . . . .	42
4.11.3.3	hreq_hdr . . . . .	42
4.11.3.4	hreq_hdr_push . . . . .	42
4.11.3.5	hreq_is_query . . . . .	42
4.11.3.6	hreq_query . . . . .	42
4.11.3.7	hreq_sign . . . . .	42
4.11.3.8	hstor_add_bucket . . . . .	42
4.11.3.9	hstor_del . . . . .	42
4.11.3.10	hstor_del_bucket . . . . .	42
4.11.3.11	hstor_free . . . . .	42
4.11.3.12	hstor_free_blist . . . . .	42
4.11.3.13	hstor_free_bucket . . . . .	42
4.11.3.14	hstor_free_keylist . . . . .	42
4.11.3.15	hstor_free_object . . . . .	42
4.11.3.16	hstor_get . . . . .	42
4.11.3.17	hstor_get_inline . . . . .	42
4.11.3.18	hstor_keys . . . . .	42
4.11.3.19	hstor_list_buckets . . . . .	42
4.11.3.20	hstor_new . . . . .	42
4.11.3.21	hstor_put . . . . .	43
4.11.3.22	hstor_put_inline . . . . .	43
4.11.3.23	hstor_set_format . . . . .	43
4.11.3.24	huri_field_escape . . . . .	43

4.11.3.25 huri_field_unescape . . . . .	43
4.11.3.26 huri_parse . . . . .	43
4.11.3.27 hutil_str2time . . . . .	43
4.11.3.28 hutil_time2str . . . . .	43
4.12 include/ncld.h File Reference . . . . .	43
4.12.1 Function Documentation . . . . .	43
4.12.1.1 ncld_close . . . . .	43
4.12.1.2 ncld_del . . . . .	44
4.12.1.3 ncld_get . . . . .	44
4.12.1.4 ncld_get_meta . . . . .	44
4.12.1.5 ncld_init . . . . .	44
4.12.1.6 ncld_open . . . . .	44
4.12.1.7 ncld_qlock . . . . .	44
4.12.1.8 ncld_read_free . . . . .	44
4.12.1.9 ncld_sess_close . . . . .	44
4.12.1.10 ncld_sess_open . . . . .	44
4.12.1.11 ncld_trylock . . . . .	44
4.12.1.12 ncld_unlock . . . . .	44
4.12.1.13 ncld_write . . . . .	44
4.13 include/objcache.h File Reference . . . . .	44
4.13.1 Macro Definition Documentation . . . . .	45
4.13.1.1 objcache_get . . . . .	45
4.13.1.2 objcache_get_dirty . . . . .	45
4.13.1.3 OC_F_DIRTY . . . . .	45
4.13.2 Function Documentation . . . . .	45
4.13.2.1 __objcache_get . . . . .	45
4.13.2.2 objcache_count . . . . .	45
4.13.2.3 objcache_fini . . . . .	45
4.13.2.4 objcache_init . . . . .	45
4.13.2.5 objcache_put . . . . .	45
4.13.2.6 objcache_test_dirty . . . . .	45
<b>Index</b>	<b>46</b>

# Chapter 1

## Data Structure Index

### 1.1 Data Structures

Here are the data structures with brief descriptions:

<a href="#">chunk_check_status</a>	5
<a href="#">chunksrv_req</a>	5
<a href="#">chunksrv_resp</a>	6
<a href="#">chunksrv_resp_chkstat</a>	7
<a href="#">chunksrv_resp_get</a>	7
<a href="#">cld_dirent_cur</a>	7
<a href="#">cld_timer</a>	8
<a href="#">cld_timer_list</a>	8
<a href="#">cldc_call_opts</a>	
Per-operation application options	9
<a href="#">cldc_fh</a>	
Open file handle associated with a session	9
<a href="#">cldc_host</a>	
Information for a single CLD server host	10
<a href="#">cldc_msg</a>	
Outgoing message, from client to server	10
<a href="#">cldc_node_metadata</a>	11
<a href="#">cldc_ops</a>	
Application-supplied facilities	12
<a href="#">cldc_pkt_info</a>	13
<a href="#">cldc_session</a>	
Single CLD client session	13
<a href="#">cldc_udp</a>	
A UDP implementation of the CLD client protocol	15
<a href="#">hail_log</a>	15
<a href="#">hstor_blist</a>	16
<a href="#">hstor_bucket</a>	16
<a href="#">hstor_client</a>	17
<a href="#">hstor_keylist</a>	17
<a href="#">hstor_object</a>	18
<a href="#">http_hdr</a>	19
<a href="#">http_req</a>	19
<a href="#">http_uri</a>	20
<a href="#">list_head</a>	21
<a href="#">nclد_fh</a>	21
<a href="#">nclد_read</a>	22
<a href="#">nclد_sess</a>	22
<a href="#">objcache</a>	23

<a href="#">objcache_entry</a>	24
<a href="#">st_client</a>	24
<a href="#">st_keylist</a>	25
<a href="#">st_object</a>	25



## Chapter 2

# File Index

### 2.1 File List

Here is a list of all files with brief descriptions:

include/ <a href="#">chunk-private.h</a> . . . . .	27
include/ <a href="#">chunk_msg.h</a> . . . . .	27
include/ <a href="#">chunkc.h</a> . . . . .	29
include/ <a href="#">chunksrv.h</a> . . . . .	31
include/ <a href="#">cld-private.h</a> . . . . .	31
include/ <a href="#">cld_common.h</a> . . . . .	32
include/ <a href="#">cldc.h</a> . . . . .	33
include/ <a href="#">elist.h</a> . . . . .	36
include/ <a href="#">hail_log.h</a> . . . . .	38
include/ <a href="#">hail_private.h</a> . . . . .	39
include/ <a href="#">hstor.h</a> . . . . .	40
include/ <a href="#">ncld.h</a> . . . . .	43
include/ <a href="#">objcache.h</a> . . . . .	44



## Chapter 3

# Data Structure Documentation

### 3.1 chunk\_check\_status Struct Reference

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

#### Data Fields

- uint8\_t [state](#)
- uint8\_t [pad](#) [3]
- uint32\_t [count](#)
- uint64\_t [lastdone](#)

#### 3.1.1 Field Documentation

3.1.1.1 uint32\_t chunk\_check\_status::count

3.1.1.2 uint64\_t chunk\_check\_status::lastdone

3.1.1.3 uint8\_t chunk\_check\_status::pad[3]

3.1.1.4 uint8\_t chunk\_check\_status::state

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

- include/[chunk\\_msg.h](#)

### 3.2 chunksrv\_req Struct Reference

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

#### Data Fields

- uint8\_t [magic](#) [[CHD\\_MAGIC\\_SZ](#)]
- uint8\_t [op](#)
- uint8\_t [flags](#)
- uint16\_t [key\\_len](#)
- uint32\_t [nonce](#)

- uint64\_t [data\\_len](#)
- char [sig](#) [[CHD\\_SIG\\_SZ](#)]

### 3.2.1 Field Documentation

3.2.1.1 uint64\_t chunksrv\_req::data\_len

3.2.1.2 uint8\_t chunksrv\_req::flags

3.2.1.3 uint16\_t chunksrv\_req::key\_len

3.2.1.4 uint8\_t chunksrv\_req::magic[[CHD\\_MAGIC\\_SZ](#)]

3.2.1.5 uint32\_t chunksrv\_req::nonce

3.2.1.6 uint8\_t chunksrv\_req::op

3.2.1.7 char chunksrv\_req::sig[[CHD\\_SIG\\_SZ](#)]

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

- include/[chunk\\_msg.h](#)

## 3.3 chunksrv\_resp Struct Reference

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

### Data Fields

- uint8\_t [magic](#) [[CHD\\_MAGIC\\_SZ](#)]
- uint8\_t [resp\\_code](#)
- uint8\_t [rsv1](#) [3]
- uint32\_t [nonce](#)
- uint64\_t [data\\_len](#)
- unsigned char [hash](#) [[CHD\\_CSUM\\_SZ](#)]

### 3.3.1 Field Documentation

3.3.1.1 uint64\_t chunksrv\_resp::data\_len

3.3.1.2 unsigned char chunksrv\_resp::hash[[CHD\\_CSUM\\_SZ](#)]

3.3.1.3 uint8\_t chunksrv\_resp::magic[[CHD\\_MAGIC\\_SZ](#)]

3.3.1.4 uint32\_t chunksrv\_resp::nonce

3.3.1.5 uint8\_t chunksrv\_resp::resp\_code

3.3.1.6 uint8\_t chunksrv\_resp::rsv1[3]

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

- include/[chunk\\_msg.h](#)

## 3.4 chunksrv\_resp\_chkstat Struct Reference

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

### Data Fields

- struct [chunksrv\\_resp](#) resp
- struct [chunk\\_check\\_status](#) chkstat

### 3.4.1 Field Documentation

3.4.1.1 struct [chunk\\_check\\_status](#) chunksrv\_resp\_chkstat::chkstat

3.4.1.2 struct [chunksrv\\_resp](#) chunksrv\_resp\_chkstat::resp

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

- include/[chunk\\_msg.h](#)

## 3.5 chunksrv\_resp\_get Struct Reference

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

### Data Fields

- struct [chunksrv\\_resp](#) resp
- [uint64\\_t](#) mtime

### 3.5.1 Field Documentation

3.5.1.1 [uint64\\_t](#) chunksrv\_resp\_get::mtime

3.5.1.2 struct [chunksrv\\_resp](#) chunksrv\_resp\_get::resp

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

- include/[chunk\\_msg.h](#)

## 3.6 cld\_dirent\_cur Struct Reference

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

### Data Fields

- const void \* p
- [size\\_t](#) tmp\_len

### 3.6.1 Field Documentation

3.6.1.1 `const void* cld_dirent_cur::p`

3.6.1.2 `size_t cld_dirent_cur::tmp_len`

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

- [include/cldc.h](#)

## 3.7 cld\_timer Struct Reference

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

### Data Fields

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

### 3.7.1 Field Documentation

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

3.7.1.2 `time_t cld_timer::expires`

3.7.1.3 `bool cld_timer::fired`

3.7.1.4 `char cld_timer::name[32]`

3.7.1.5 `bool cld_timer::on_list`

3.7.1.6 `void* cld_timer::userdata`

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

- [include/cld\\_common.h](#)

## 3.8 cld\_timer\_list Struct Reference

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

### Data Fields

- `GList * list`
- `time_t runmark`

### 3.8.1 Field Documentation

3.8.1.1 `GList*` `cld_timer_list::list`

3.8.1.2 `time_t` `cld_timer_list::runmark`

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

- `include/cld_common.h`

## 3.9 cldc\_call\_opts Struct Reference

per-operation application options

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

### Data Fields

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

### 3.9.1 Detailed Description

per-operation application options

### 3.9.2 Field Documentation

3.9.2.1 `int(* cldc_call_opts::cb)(struct cldc_call_opts *, enum cle_err_codes)`

3.9.2.2 `void*` `cldc_call_opts::private`

3.9.2.3 `struct cld_msg_get_resp` `cldc_call_opts::resp`

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

- `include/cldc.h`

## 3.10 cldc\_fh Struct Reference

an open file handle associated with a session

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

### Data Fields

- `uint64_t` `fh`
- `struct cldc_session *` `sess`
- `bool` `valid`

### 3.10.1 Detailed Description

an open file handle associated with a session

### 3.10.2 Field Documentation

3.10.2.1 `uint64_t cldc_fh::fh`

3.10.2.2 `struct cldc_session* cldc_fh::sess`

3.10.2.3 `bool cldc_fh::valid`

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

- [include/cldc.h](#)

## 3.11 cldc\_host Struct Reference

Information for a single CLD server host.

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

### Data Fields

- unsigned int [prio](#)
- unsigned int [weight](#)
- char \* [host](#)
- unsigned short [port](#)

### 3.11.1 Detailed Description

Information for a single CLD server host.

### 3.11.2 Field Documentation

3.11.2.1 `char* cldc_host::host`

3.11.2.2 `unsigned short cldc_host::port`

3.11.2.3 `unsigned int cldc_host::prio`

3.11.2.4 `unsigned int cldc_host::weight`

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

- [include/cldc.h](#)

## 3.12 cldc\_msg Struct Reference

an outgoing message, from client to server

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



## Data Fields

- uint64\_t [xid](#)
- enum cld\_msg\_op [op](#)
- struct [cldc\\_session](#) \* [sess](#)
- ssize\_t(\* [cb](#))(struct [cldc\\_msg](#) \*, const void \*, size\_t, enum cle\_err\_codes)
- void \* [cb\\_private](#)
- struct [cldc\\_call\\_opts](#) [copts](#)
- bool [done](#)
- time\_t [expire\\_time](#)
- int [n\\_pkts](#)
- struct [cldc\\_pkt\\_info](#) \* [pkt\\_info](#) [0]

### 3.12.1 Detailed Description

an outgoing message, from client to server

### 3.12.2 Field Documentation

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

3.12.2.2 `void* cldc_msg::cb_private`

3.12.2.3 `struct cldc_call_opts cldc_msg::copts`

3.12.2.4 `bool cldc_msg::done`

3.12.2.5 `time_t cldc_msg::expire_time`

3.12.2.6 `int cldc_msg::n_pkts`

3.12.2.7 `enum cld_msg_op cldc_msg::op`

3.12.2.8 `struct cldc_pkt_info* cldc_msg::pkt_info[0]`

3.12.2.9 `struct cldc_session* cldc_msg::sess`

3.12.2.10 `uint64_t cldc_msg::xid`

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

- [include/cldc.h](#)

## 3.13 cldc\_node\_metadata Struct Reference

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

## Data Fields

- quad\_t [inum](#)
- quad\_t [vers](#)
- quad\_t [time\\_create](#)

- `quad_t time_modify`
- `int flags`
- `const char * inode_name`

### 3.13.1 Field Documentation

3.13.1.1 `int cldc_node_metadata::flags`

3.13.1.2 `const char* cldc_node_metadata::inode_name`

3.13.1.3 `quad_t cldc_node_metadata::inum`

3.13.1.4 `quad_t cldc_node_metadata::time_create`

3.13.1.5 `quad_t cldc_node_metadata::time_modify`

3.13.1.6 `quad_t cldc_node_metadata::vers`

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

- `include/cldc.h`

## 3.14 cldc\_ops Struct Reference

application-supplied facilities

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

### Data Fields

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

### 3.14.1 Detailed Description

application-supplied facilities

### 3.14.2 Field Documentation

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

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

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

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

- `include/cldc.h`

## 3.15 cldc\_pkt\_info Struct Reference

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

### Data Fields

- int [pkt\\_len](#)
- int [hdr\\_len](#)
- int [retries](#)
- char [user](#) [CLD\_MAX\_USERNAME]
- char [data](#) [0]

### 3.15.1 Field Documentation

3.15.1.1 char cldc\_pkt\_info::data[0]

3.15.1.2 int cldc\_pkt\_info::hdr\_len

3.15.1.3 int cldc\_pkt\_info::pkt\_len

3.15.1.4 int cldc\_pkt\_info::retries

3.15.1.5 char cldc\_pkt\_info::user[CLD\_MAX\_USERNAME]

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

- include/[cldc.h](#)

## 3.16 cldc\_session Struct Reference

a single CLD client session

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

### Data Fields

- uint8\_t [sid](#) [CLD\_SID\_SZ]
- const struct [cldc\\_ops](#) \* [ops](#)
- struct [hail\\_log](#) [log](#)
- void \* [private](#)
- uint8\_t [addr](#) [64]
- size\_t [addr\\_len](#)
- GList \* [cfh](#)
- GList \* [out\\_msg](#)
- time\_t [msg\\_scan\\_time](#)
- time\_t [expire\\_time](#)
- bool [expired](#)
- uint64\_t [next\\_seqid\\_in](#)
- uint64\_t [next\\_seqid\\_in\\_tr](#)
- uint64\_t [next\\_seqid\\_out](#)
- char [user](#) [CLD\_MAX\_USERNAME]
- char [secret\\_key](#) [CLD\_MAX\_SECRET\_KEY]

- bool `confirmed`
- enum `cld_msg_op` `msg_buf_op`
- unsigned int `msg_buf_len`
- char `msg_buf` [CLD\_MAX\_MSG\_SZ]
- char `payload` [CLD\_MAX\_PAYLOAD\_SZ]
- char `inode_name_temp` [CLD\_INODE\_NAME\_MAX]

### 3.16.1 Detailed Description

a single CLD client session

### 3.16.2 Field Documentation

- 3.16.2.1 `uint8_t cldc_session::addr[64]`
- 3.16.2.2 `size_t cldc_session::addr_len`
- 3.16.2.3 `GList* cldc_session::cfh`
- 3.16.2.4 `bool cldc_session::confirmed`
- 3.16.2.5 `time_t cldc_session::expire_time`
- 3.16.2.6 `bool cldc_session::expired`
- 3.16.2.7 `char cldc_session::inode_name_temp[CLD_INODE_NAME_MAX]`
- 3.16.2.8 `struct hail_log cldc_session::log`
- 3.16.2.9 `char cldc_session::msg_buf[CLD_MAX_MSG_SZ]`
- 3.16.2.10 `unsigned int cldc_session::msg_buf_len`
- 3.16.2.11 `enum cld_msg_op cldc_session::msg_buf_op`
- 3.16.2.12 `time_t cldc_session::msg_scan_time`
- 3.16.2.13 `uint64_t cldc_session::next_seqid_in`
- 3.16.2.14 `uint64_t cldc_session::next_seqid_in_tr`
- 3.16.2.15 `uint64_t cldc_session::next_seqid_out`
- 3.16.2.16 `const struct cldc_ops* cldc_session::ops`
- 3.16.2.17 `GList* cldc_session::out_msg`
- 3.16.2.18 `char cldc_session::payload[CLD_MAX_PAYLOAD_SZ]`
- 3.16.2.19 `void* cldc_session::private`
- 3.16.2.20 `char cldc_session::secret_key[CLD_MAX_SECRET_KEY]`
- 3.16.2.21 `uint8_t cldc_session::sid[CLD_SID_SZ]`

## 3.16.2.22 char cldc\_session::user[CLD\_MAX\_USERNAME]

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

- include/cldc.h

## 3.17 cldc\_udp Struct Reference

A UDP implementation of the CLD client protocol.

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

### Data Fields

- uint8\_t [addr](#) [64]
- size\_t [addr\\_len](#)
- int [fd](#)
- struct [cldc\\_session](#) \* [sess](#)
- int(\* [cb](#) )(struct [cldc\\_session](#) \*, void \*)
- void \* [cb\\_private](#)

### 3.17.1 Detailed Description

A UDP implementation of the CLD client protocol.

### 3.17.2 Field Documentation

3.17.2.1 uint8\_t cldc\_udp::addr[64]

3.17.2.2 size\_t cldc\_udp::addr\_len

3.17.2.3 int(\* cldc\_udp::cb)(struct [cldc\\_session](#) \*, void \*)

3.17.2.4 void\* cldc\_udp::cb\_private

3.17.2.5 int cldc\_udp::fd

3.17.2.6 struct [cldc\\_session](#)\* cldc\_udp::sess

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

- include/cldc.h

## 3.18 hail\_log Struct Reference

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

### Data Fields

- void(\* [func](#) )(int prio, const char \*fmt,...) [ATTR\\_PRINTF](#)(2
- void(\*) boo [debug](#) )
- bool [verbose](#)

### 3.18.1 Field Documentation

3.18.1.1 void(\*) boo hail\_log::debug)

3.18.1.2 void(\* hail\_log::func)(int prio, const char \*fmt,...) ATTR\_PRINTF(2

3.18.1.3 bool hail\_log::verbose

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

- include/[hail\\_log.h](#)

## 3.19 hstor\_blist Struct Reference

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

### Data Fields

- char \* [own\\_id](#)
- char \* [own\\_name](#)
- GList \* [list](#)

### 3.19.1 Field Documentation

3.19.1.1 GList\* hstor\_blist::list

3.19.1.2 char\* hstor\_blist::own\_id

3.19.1.3 char\* hstor\_blist::own\_name

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

- include/[hstor.h](#)

## 3.20 hstor\_bucket Struct Reference

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

### Data Fields

- char \* [name](#)
- char \* [time\\_create](#)

### 3.20.1 Field Documentation

3.20.1.1 char\* hstor\_bucket::name

3.20.1.2 char\* hstor\_bucket::time\_create

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

- include/[hstor.h](#)

## 3.21 hstor\_client Struct Reference

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

### Data Fields

- CURL \* [curl](#)
- char \* [acc](#)
- char \* [host](#)
- char \* [user](#)
- char \* [key](#)
- bool [verbose](#)
- bool [subdomain](#)

### 3.21.1 Field Documentation

3.21.1.1 char\* hstor\_client::acc

3.21.1.2 CURL\* hstor\_client::curl

3.21.1.3 char\* hstor\_client::host

3.21.1.4 char\* hstor\_client::key

3.21.1.5 bool hstor\_client::subdomain

3.21.1.6 char\* hstor\_client::user

3.21.1.7 bool hstor\_client::verbose

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

- include/[hstor.h](#)

## 3.22 hstor\_keylist Struct Reference

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

### Data Fields

- char \* [name](#)
- char \* [prefix](#)
- char \* [marker](#)
- char \* [delim](#)
- unsigned int [max\\_keys](#)
- bool [trunc](#)
- GList \* [contents](#)
- GList \* [common\\_pfx](#)

### 3.22.1 Field Documentation

3.22.1.1 `GList* hstor_keylist::common_pfx`

3.22.1.2 `GList* hstor_keylist::contents`

3.22.1.3 `char* hstor_keylist::delim`

3.22.1.4 `char* hstor_keylist::marker`

3.22.1.5 `unsigned int hstor_keylist::max_keys`

3.22.1.6 `char* hstor_keylist::name`

3.22.1.7 `char* hstor_keylist::prefix`

3.22.1.8 `bool hstor_keylist::trunc`

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

- [include/hstor.h](#)

## 3.23 hstor\_object Struct Reference

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

### Data Fields

- `char * key`
- `char * time\_mod`
- `char * etag`
- `uint64_t size`
- `char * storage`
- `char * own\_id`
- `char * own\_name`

### 3.23.1 Field Documentation

3.23.1.1 `char* hstor_object::etag`

3.23.1.2 `char* hstor_object::key`

3.23.1.3 `char* hstor_object::own_id`

3.23.1.4 `char* hstor_object::own_name`

3.23.1.5 `uint64_t hstor_object::size`

3.23.1.6 `char* hstor_object::storage`

3.23.1.7 `char* hstor_object::time_mod`

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



- include/[hstor.h](#)

## 3.24 http\_hdr Struct Reference

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

### Data Fields

- char \* [key](#)
- char \* [val](#)

#### 3.24.1 Field Documentation

3.24.1.1 char\* http\_hdr::key

3.24.1.2 char\* http\_hdr::val

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

- include/[hstor.h](#)

## 3.25 http\_req Struct Reference

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

### Data Fields

- char \* [method](#)
- struct [http\\_uri](#) uri
- int [major](#)
- int [minor](#)
- char \* [orig\\_path](#)
- unsigned int [n\\_hdr](#)
- struct [http\\_hdr](#) hdr [[HREQ\\_MAX\\_HDR](#)]

#### 3.25.1 Field Documentation

3.25.1.1 struct http\_hdr http\_req::hdr[HREQ\_MAX\_HDR]

3.25.1.2 int http\_req::major

3.25.1.3 char\* http\_req::method

3.25.1.4 int http\_req::minor

3.25.1.5 unsigned int http\_req::n\_hdr

3.25.1.6 char\* http\_req::orig\_path

### 3.25.1.7 struct http\_uri http\_req::uri

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

- include/hstor.h

## 3.26 http\_uri Struct Reference

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

### Data Fields

- char \* [scheme](#)
- unsigned int [scheme\\_len](#)
- char \* [userinfo](#)
- unsigned int [userinfo\\_len](#)
- char \* [hostname](#)
- unsigned int [hostname\\_len](#)
- unsigned int [port](#)
- char \* [path](#)
- unsigned int [path\\_len](#)
- char \* [query](#)
- unsigned int [query\\_len](#)
- char \* [fragment](#)
- unsigned int [fragment\\_len](#)

### 3.26.1 Field Documentation

3.26.1.1 char\* http\_uri::fragment

3.26.1.2 unsigned int http\_uri::fragment\_len

3.26.1.3 char\* http\_uri::hostname

3.26.1.4 unsigned int http\_uri::hostname\_len

3.26.1.5 char\* http\_uri::path

3.26.1.6 unsigned int http\_uri::path\_len

3.26.1.7 unsigned int http\_uri::port

3.26.1.8 char\* http\_uri::query

3.26.1.9 unsigned int http\_uri::query\_len

3.26.1.10 char\* http\_uri::scheme

3.26.1.11 unsigned int http\_uri::scheme\_len

3.26.1.12 char\* http\_uri::userinfo

## 3.26.1.13 unsigned int http\_uri::userinfo\_len

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

- [include/hstor.h](#)

## 3.27 list\_head Struct Reference

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

### Data Fields

- struct [list\\_head](#) \* [next](#)
- struct [list\\_head](#) \* [prev](#)

### 3.27.1 Field Documentation

## 3.27.1.1 struct list\_head\* list\_head::next

## 3.27.1.2 struct list\_head \* list\_head::prev

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

- [include/elist.h](#)

## 3.28 ncld\_fh Struct Reference

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

### Data Fields

- struct [ncld\\_sess](#) \* [sess](#)
- struct [cldc\\_fh](#) \* [fh](#)
- bool [is\\_open](#)
- int [errc](#)
- int [nios](#)
- unsigned int [event\\_mask](#)
- void(\* [event\\_func](#))(void \*, unsigned int)
- void \* [event\\_arg](#)

### 3.28.1 Field Documentation

## 3.28.1.1 int ncld\_fh::errc

## 3.28.1.2 void\* ncld\_fh::event\_arg

## 3.28.1.3 void(\* ncld\_fh::event\_func)(void \*, unsigned int)

## 3.28.1.4 unsigned int ncld\_fh::event\_mask

3.28.1.5 `struct cldc_fh* 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 [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`

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`

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 nclد\_sess Struct Reference

```
#include <nclد.h>
```

### Data Fields

- char \* [host](#)
- unsigned short [port](#)

- GMutex \* [mutex](#)
- GCond \* [cond](#)
- GThread \* [thread](#)
- bool [is\\_up](#)
- bool [open\\_done](#)
- int [errc](#)
- GList \* [handles](#)
- int [to\\_thread](#) [2]
- struct [cldc\\_udp](#) \* [udp](#)
- struct [cld\\_timer](#) [udp\\_timer](#)
- struct [cld\\_timer\\_list](#) [tlist](#)
- void(\* [event](#) )(void \*, unsigned int)
- void \* [event\\_arg](#)

### 3.30.1 Field Documentation

3.30.1.1 GCond\* [ncld\\_sess::cond](#)

3.30.1.2 int [ncld\\_sess::errc](#)

3.30.1.3 void(\* [ncld\\_sess::event](#) )(void \*, unsigned int)

3.30.1.4 void\* [ncld\\_sess::event\\_arg](#)

3.30.1.5 GList\* [ncld\\_sess::handles](#)

3.30.1.6 char\* [ncld\\_sess::host](#)

3.30.1.7 bool [ncld\\_sess::is\\_up](#)

3.30.1.8 GMutex\* [ncld\\_sess::mutex](#)

3.30.1.9 bool [ncld\\_sess::open\\_done](#)

3.30.1.10 unsigned short [ncld\\_sess::port](#)

3.30.1.11 GThread\* [ncld\\_sess::thread](#)

3.30.1.12 struct [cld\\_timer\\_list](#) [ncld\\_sess::tlist](#)

3.30.1.13 int [ncld\\_sess::to\\_thread](#)[2]

3.30.1.14 struct [cldc\\_udp](#)\* [ncld\\_sess::udp](#)

3.30.1.15 struct [cld\\_timer](#) [ncld\\_sess::udp\\_timer](#)

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

- [include/ncld.h](#)

## 3.31 objcache Struct Reference

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

## Data Fields

- GMutex \* [lock](#)
- GHashTable \* [table](#)

### 3.31.1 Field Documentation

3.31.1.1 GMutex\* objcache::lock

3.31.1.2 GHashTable\* objcache::table

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

- include/[objcache.h](#)

## 3.32 objcache\_entry Struct Reference

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

## Data Fields

- unsigned int [hash](#)
- unsigned int [flags](#)
- int [ref](#)

### 3.32.1 Field Documentation

3.32.1.1 unsigned int objcache\_entry::flags

3.32.1.2 unsigned int objcache\_entry::hash

3.32.1.3 int objcache\_entry::ref

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

- include/[objcache.h](#)

## 3.33 st\_client Struct Reference

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

## Data Fields

- char \* [host](#)
- char \* [user](#)
- char \* [key](#)
- bool [verbose](#)
- int [fd](#)
- SSL\_CTX \* [ssl\\_ctx](#)
- SSL \* [ssl](#)
- char [req\\_buf](#) [sizeof(struct chunksrv\_req)+CHD\_KEY\_SZ]

### 3.33.1 Field Documentation

3.33.1.1 int st\_client::fd

3.33.1.2 char\* st\_client::host

3.33.1.3 char\* st\_client::key

3.33.1.4 char st\_client::req\_buf[sizeof(struct chunksrv\_req)+CHD\_KEY\_SZ]

3.33.1.5 SSL\* st\_client::ssl

3.33.1.6 SSL\_CTX\* st\_client::ssl\_ctx

3.33.1.7 char\* st\_client::user

3.33.1.8 bool st\_client::verbose

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

- include/[chunkc.h](#)

## 3.34 st\_keylist Struct Reference

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

### Data Fields

- char \* [name](#)
- GList \* [contents](#)

### 3.34.1 Field Documentation

3.34.1.1 GList\* st\_keylist::contents

3.34.1.2 char\* st\_keylist::name

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

- include/[chunkc.h](#)

## 3.35 st\_object Struct Reference

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

### Data Fields

- char \* [name](#)
- char \* [time\\_mod](#)
- char \* [etag](#)
- uint64\_t [size](#)
- char \* [owner](#)

### 3.35.1 Field Documentation

3.35.1.1 `char* st_object::etag`

3.35.1.2 `char* st_object::name`

3.35.1.3 `char* st_object::owner`

3.35.1.4 `uint64_t st_object::size`

3.35.1.5 `char* st_object::time_mod`

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

- [include/chunkc.h](#)



## Chapter 4

# File Documentation

### 4.1 include/chunk-private.h File Reference

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

#### Macros

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

#### 4.1.1 Macro Definition Documentation

4.1.1.1 `#define BAD_TPATH_FMT "%s/bad"`

4.1.1.2 `#define MDB_TPATH_FMT "%s/%X"`

4.1.1.3 `#define PREFIX_LEN 3`

### 4.2 include/chunk\_msg.h File Reference

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

#### Data Structures

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

#### Macros

- `#define CHUNKD_MAGIC "CHUNKDv1"`

## Enumerations

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

### 4.2.1 Macro Definition Documentation

4.2.1.1 `#define CHUNKD_MAGIC "CHUNKDv1"`

### 4.2.2 Enumeration Type Documentation

4.2.2.1 anonymous enum

Enumerator

***CHD\_MAGIC\_SZ***  
***CHD\_USER\_SZ***  
***CHD\_KEY\_SZ***  
***CHD\_CSUM\_SZ***  
***CHD\_SIG\_SZ***

4.2.2.2 enum `chunk_check_state`

Enumerator

***chk\_Off***  
***chk\_Idle***  
***chk\_Active***

4.2.2.3 enum `chunk_errcode`

Enumerator

***che\_Success***  
***che\_AccessDenied***  
***che\_InternalError***  
***che\_InvalidArgument***  
***che\_InvalidURI***  
***che\_NoSuchKey***  
***che\_SignatureDoesNotMatch***  
***che\_InvalidKey***

***che\_InvalidTable***  
***che\_Busy***  
***che\_KeyExists***

#### 4.2.2.4 enum chunk\_flags

Enumerator

***CHF\_SYNC***  
***CHF\_TBL\_CREAT***  
***CHF\_TBL\_EXCL***

#### 4.2.2.5 enum chunksrv\_ops

Enumerator

***CHO\_NOP***  
***CHO\_GET***  
***CHO\_GET\_META***  
***CHO\_PUT***  
***CHO\_DEL***  
***CHO\_LIST***  
***CHO\_LOGIN***  
***CHO\_TABLE\_OPEN***  
***CHO\_CHECK\_START***  
***CHO\_CHECK\_STATUS***  
***CHO\_START\_TLS***  
***CHO\_CP***

## 4.3 include/chunkc.h File Reference

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

### Data Structures

- struct [st\\_object](#)
- struct [st\\_keylist](#)
- struct [st\\_client](#)

## Functions

- void [stc\\_free](#) (struct [st\\_client](#) \*stc)
- void [stc\\_free\\_keylist](#) (struct [st\\_keylist](#) \*keylist)
- void [stc\\_free\\_object](#) (struct [st\\_object](#) \*obj)
- void [stc\\_init](#) (void)
- struct [st\\_client](#) \* [stc\\_new](#) (const char \*service\_host, int port, const char \*user, const char \*secret\_key, bool encrypt)
- bool [stc\\_table\\_open](#) (struct [st\\_client](#) \*stc, const void \*key, size\_t key\_len, uint32\_t flags)
- bool [stc\\_get](#) (struct [st\\_client](#) \*stc, const void \*key, size\_t key\_len, size\_t(\*write\_cb)(void \*, size\_t, size\_t, void \*), void \*user\_data)
- void \* [stc\\_get\\_inline](#) (struct [st\\_client](#) \*stc, const void \*key, size\_t key\_len, size\_t \*len)
- bool [stc\\_get\\_start](#) (struct [st\\_client](#) \*stc, const void \*key, size\_t key\_len, int \*pfd, uint64\_t \*len)
- size\_t [stc\\_get\\_recv](#) (struct [st\\_client](#) \*stc, void \*data, size\_t len)
- bool [stc\\_put](#) (struct [st\\_client](#) \*stc, const void \*key, size\_t key\_len, size\_t(\*read\_cb)(void \*, size\_t, size\_t, void \*), uint64\_t len, void \*user\_data, uint32\_t flags)
- bool [stc\\_put\\_start](#) (struct [st\\_client](#) \*stc, const void \*key, size\_t key\_len, uint64\_t cont\_len, int \*pfd, uint32\_t flags)
- size\_t [stc\\_put\\_send](#) (struct [st\\_client](#) \*stc, void \*data, size\_t len)
- bool [stc\\_put\\_sync](#) (struct [st\\_client](#) \*stc)
- bool [stc\\_put\\_inline](#) (struct [st\\_client](#) \*stc, const void \*key, size\_t key\_len, void \*data, uint64\_t len, uint32\_t flags)
- bool [stc\\_cp](#) (struct [st\\_client](#) \*stc, const void \*dest\_key, size\_t dest\_key\_len, const void \*src\_key, size\_t src\_key\_len)
- bool [stc\\_del](#) (struct [st\\_client](#) \*stc, const void \*key, size\_t key\_len)
- bool [stc\\_ping](#) (struct [st\\_client](#) \*stc)
- bool [stc\\_check\\_start](#) (struct [st\\_client](#) \*stc)
- bool [stc\\_check\\_status](#) (struct [st\\_client](#) \*stc, struct [chunk\\_check\\_status](#) \*out)
- struct [st\\_keylist](#) \* [stc\\_keys](#) (struct [st\\_client](#) \*stc)
- int [stc\\_readport](#) (const char \*fname)

### 4.3.1 Function Documentation

4.3.1.1 bool [stc\\_check\\_start](#) ( struct [st\\_client](#) \* *stc* )

4.3.1.2 bool [stc\\_check\\_status](#) ( struct [st\\_client](#) \* *stc*, struct [chunk\\_check\\_status](#) \* *out* )

4.3.1.3 bool [stc\\_cp](#) ( struct [st\\_client](#) \* *stc*, const void \* *dest\_key*, size\_t *dest\_key\_len*, const void \* *src\_key*, size\_t *src\_key\_len* )

4.3.1.4 bool [stc\\_del](#) ( struct [st\\_client](#) \* *stc*, const void \* *key*, size\_t *key\_len* )

4.3.1.5 void [stc\\_free](#) ( struct [st\\_client](#) \* *stc* )

4.3.1.6 void [stc\\_free\\_keylist](#) ( struct [st\\_keylist](#) \* *keylist* )

4.3.1.7 void [stc\\_free\\_object](#) ( struct [st\\_object](#) \* *obj* )

4.3.1.8 bool [stc\\_get](#) ( struct [st\\_client](#) \* *stc*, const void \* *key*, size\_t *key\_len*, size\_t(\*)(void \*, size\_t, size\_t, void \*) *write\_cb*, void \* *user\_data* )

4.3.1.9 void\* [stc\\_get\\_inline](#) ( struct [st\\_client](#) \* *stc*, const void \* *key*, size\_t *key\_len*, size\_t \* *len* )

4.3.1.10 size\_t [stc\\_get\\_recv](#) ( struct [st\\_client](#) \* *stc*, void \* *data*, size\_t *len* )

- 4.3.1.11 `bool stc_get_start ( struct st_client * stc, const void * key, size_t key_len, int * pfd, uint64_t * len )`
- 4.3.1.12 `void stc_init ( void )`
- 4.3.1.13 `struct st_keylist* stc_keys ( struct st_client * stc )`
- 4.3.1.14 `struct st_client* stc_new ( const char * service_host, int port, const char * user, const char * secret_key, bool encrypt )`
- 4.3.1.15 `bool stc_ping ( struct st_client * stc )`
- 4.3.1.16 `bool stc_put ( struct st_client * stc, const void * key, size_t key_len, size_t(*)(void *, size_t, size_t, void *) read_cb, uint64_t len, void * user_data, uint32_t flags )`
- 4.3.1.17 `bool stc_put_inline ( struct st_client * stc, const void * key, size_t key_len, void * data, uint64_t len, uint32_t flags )`
- 4.3.1.18 `size_t stc_put_send ( struct st_client * stc, void * data, size_t len )`
- 4.3.1.19 `bool stc_put_start ( struct st_client * stc, const void * key, size_t key_len, uint64_t cont_len, int * pfd, uint32_t flags )`
- 4.3.1.20 `bool stc_put_sync ( struct st_client * stc )`
- 4.3.1.21 `int stc_readport ( const char * fname )`
- 4.3.1.22 `bool stc_table_open ( struct st_client * stc, const void * key, size_t key_len, uint32_t flags )`

## 4.4 include/chunksrv.h File Reference

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

### Functions

- `size_t req_len` (const struct [chunksrv\\_req](#) \*req)
- `void chreq_sign` (struct [chunksrv\\_req](#) \*req, const char \*key, char \*b64hmac\_out)

#### 4.4.1 Function Documentation

- 4.4.1.1 `void chreq_sign ( struct chunksrv_req * req, const char * key, char * b64hmac_out )`
- 4.4.1.2 `size_t req_len ( const struct chunksrv_req * req )`

## 4.5 include/cld-private.h File Reference

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

## 4.6 include/cld\_common.h File Reference

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

### Data Structures

- struct [cld\\_timer](#)
- struct [cld\\_timer\\_list](#)

### Macros

- #define [CLD\\_ALIGN8](#)(n) ((8 - ((n) & 7)) & 7)
- #define [SIDFMT](#) "%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 Macro Definition Documentation

4.6.1.1 #define [CLD\\_ALIGN8](#)( n ) ((8 - ((n) & 7)) & 7)

4.6.1.2 #define [CLD\\_PKT\\_FTR\\_LEN](#) sizeof(struct cld\_pkt\_ftr)

Length of the packet footer.

This size is fixed

4.6.1.3 `#define PKT_HDR_TO_STR_SCRATCH_LEN 128`

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

4.6.1.5 `#define SIDFMT "%016llx"`

## 4.6.2 Function Documentation

4.6.2.1 `struct __attribute__ ( (packed) )`

Footer that appears at the end of each packet.

< packet sequence ID

< packet signature

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

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

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

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

4.6.2.6 `const char* cld_opstr ( enum cld_msg_op )`

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

4.6.2.8 `void cld_rand64 ( void * p )`

4.6.2.9 `int cld_readport ( const char * fname )`

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

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

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

4.6.2.13 `time_t cld_timers_run ( struct cld_timer_list * tlist )`

## 4.7 include/cldc.h File Reference

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

## Data Structures

- struct [cldc\\_call\\_opts](#)  
*per-operation application options*
- struct [cldc\\_node\\_metadata](#)

- struct [cldc\\_pkt\\_info](#)
- struct [cldc\\_msg](#)  
*an outgoing message, from client to server*
- struct [cldc\\_fh](#)  
*an open file handle associated with a session*
- struct [cldc\\_ops](#)  
*application-supplied facilities*
- struct [cldc\\_session](#)  
*a single CLD client session*
- struct [cldc\\_host](#)  
*Information for a single CLD server host.*
- struct [cldc\\_udp](#)  
*A UDP implementation of the CLD client protocol.*
- struct [cld\\_dirent\\_cur](#)

## Functions

- int [cldc\\_receive\\_pkt](#) (struct [cldc\\_session](#) \*sess, const void \*net\_addr, size\_t net\_addrlen, const void \*buf, size\_t buflen)  
*Packet received from remote host.*
- void [cldc\\_init](#) (void)
- int [cldc\\_new\\_sess](#) (const struct [cldc\\_ops](#) \*ops, const struct [cldc\\_call\\_opts](#) \*copts, const void \*addr, size\_t addr\_len, const char \*user, const char \*secret\_key, void \*private, struct [cldc\\_session](#) \*\*sess\_out)
- void [cldc\\_kill\\_sess](#) (struct [cldc\\_session](#) \*sess)
- int [cldc\\_end\\_sess](#) (struct [cldc\\_session](#) \*sess, const struct [cldc\\_call\\_opts](#) \*copts)
- int [cldc\\_nop](#) (struct [cldc\\_session](#) \*sess, const struct [cldc\\_call\\_opts](#) \*copts)
- int [cldc\\_del](#) (struct [cldc\\_session](#) \*sess, const struct [cldc\\_call\\_opts](#) \*copts, const char \*pathname)
- int [cldc\\_open](#) (struct [cldc\\_session](#) \*sess, const struct [cldc\\_call\\_opts](#) \*copts, const char \*pathname, uint32\_t open\_mode, uint32\_t events, struct [cldc\\_fh](#) \*\*fh\_out)
- int [cldc\\_close](#) (struct [cldc\\_fh](#) \*fh, const struct [cldc\\_call\\_opts](#) \*copts)
- int [cldc\\_unlock](#) (struct [cldc\\_fh](#) \*fh, const struct [cldc\\_call\\_opts](#) \*copts)
- int [cldc\\_lock](#) (struct [cldc\\_fh](#) \*fh, const struct [cldc\\_call\\_opts](#) \*copts, uint32\_t lock\_flags, bool wait\_for\_lock)
- int [cldc\\_put](#) (struct [cldc\\_fh](#) \*fh, const struct [cldc\\_call\\_opts](#) \*copts, const void \*data, size\_t data\_len)
- int [cldc\\_get](#) (struct [cldc\\_fh](#) \*fh, const struct [cldc\\_call\\_opts](#) \*copts, bool metadata\_only)
- int [cldc\\_dirent\\_count](#) (const void \*data, size\_t data\_len)
- int [cldc\\_dirent\\_first](#) (struct [cld\\_dirent\\_cur](#) \*dc)
- int [cldc\\_dirent\\_next](#) (struct [cld\\_dirent\\_cur](#) \*dc)
- void [cldc\\_dirent\\_cur\\_init](#) (struct [cld\\_dirent\\_cur](#) \*dc, const void \*buf, size\_t buflen)
- void [cldc\\_dirent\\_cur\\_fini](#) (struct [cld\\_dirent\\_cur](#) \*dc)
- char \* [cldc\\_dirent\\_name](#) (struct [cld\\_dirent\\_cur](#) \*dc)
- void [cldc\\_copts\\_get\\_data](#) (const struct [cldc\\_call\\_opts](#) \*copts, char \*\*data, size\_t \*data\_len)
- void [cldc\\_copts\\_get\\_metadata](#) (const struct [cldc\\_call\\_opts](#) \*copts, struct [cldc\\_node\\_metadata](#) \*md)
- void [cldc\\_udp\\_free](#) (struct [cldc\\_udp](#) \*udp)
- int [cldc\\_udp\\_new](#) (const char \*hostname, int port, struct [cldc\\_udp](#) \*\*udp\_out)
- int [cldc\\_udp\\_receive\\_pkt](#) (struct [cldc\\_udp](#) \*udp)
- int [cldc\\_udp\\_pkt\\_send](#) (void \*private, const void \*addr, size\_t addrlen, const void \*buf, size\_t buflen)
- int [cldc\\_getaddr](#) (GList \*\*host\_list, const char \*thishost, struct [hail\\_log](#) \*log)
- int [cldc\\_saveaddr](#) (struct [cldc\\_host](#) \*hp, unsigned int priority, unsigned int weight, unsigned int port, unsigned int nlen, const char \*name, struct [hail\\_log](#) \*log)



### 4.7.1 Function Documentation

- 4.7.1.1 `int cldc_close ( struct cldc_fh * fh, const struct cldc_call_opts * copts )`
- 4.7.1.2 `void cldc_copts_get_data ( const struct cldc_call_opts * copts, char ** data, size_t * data_len )`
- 4.7.1.3 `void cldc_copts_get_metadata ( const struct cldc_call_opts * copts, struct cldc_node_metadata * md )`
- 4.7.1.4 `int cldc_del ( struct cldc_session * sess, const struct cldc_call_opts * copts, const char * pathname )`
- 4.7.1.5 `int cldc_dirent_count ( const void * data, size_t data_len )`
- 4.7.1.6 `void cldc_dirent_cur_fini ( struct cld_dirent_cur * dc )`
- 4.7.1.7 `void cldc_dirent_cur_init ( struct cld_dirent_cur * dc, const void * buf, size_t buflen )`
- 4.7.1.8 `int cldc_dirent_first ( struct cld_dirent_cur * dc )`
- 4.7.1.9 `char* cldc_dirent_name ( struct cld_dirent_cur * dc )`
- 4.7.1.10 `int cldc_dirent_next ( struct cld_dirent_cur * dc )`
- 4.7.1.11 `int cldc_end_sess ( struct cldc_session * sess, const struct cldc_call_opts * copts )`
- 4.7.1.12 `int cldc_get ( struct cldc_fh * fh, const struct cldc_call_opts * copts, bool metadata_only )`
- 4.7.1.13 `int cldc_getaddr ( GList ** host_list, const char * thishost, struct hail_log * log )`
- 4.7.1.14 `void cldc_init ( void )`
- 4.7.1.15 `void cldc_kill_sess ( struct cldc_session * sess )`
- 4.7.1.16 `int cldc_lock ( struct cldc_fh * fh, const struct cldc_call_opts * copts, uint32_t lock_flags, bool wait_for_lock )`
- 4.7.1.17 `int cldc_new_sess ( const struct cldc_ops * ops, const struct cldc_call_opts * copts, const void * addr, size_t addr_len, const char * user, const char * secret_key, void * private, struct cldc_session ** sess_out )`
- 4.7.1.18 `int cldc_nop ( struct cldc_session * sess, const struct cldc_call_opts * copts )`
- 4.7.1.19 `int cldc_open ( struct cldc_session * sess, const struct cldc_call_opts * copts, const char * pathname, uint32_t open_mode, uint32_t events, struct cldc_fh ** fh_out )`
- 4.7.1.20 `int cldc_put ( struct cldc_fh * fh, const struct cldc_call_opts * copts, const void * data, size_t data_len )`
- 4.7.1.21 `int cldc_receive_pkt ( struct cldc_session * sess, const void * net_addr, size_t net_addrlen, const void * buf, size_t buflen )`

Packet received from remote host.

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

#### Parameters

<i>sess</i>	Session associated with received packet
-------------	---

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

## Returns

Zero for success, non-zero on error

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

4.7.1.23 `void cldc_udp_free ( struct cldc_udp * udp )`

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

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

4.7.1.26 `int cldc_udp_receive_pkt ( struct cldc_udp * udp )`

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

## 4.8 include/elist.h File Reference

### Data Structures

- struct [list\\_head](#)

### Macros

- `#define LIST_HEAD_INIT(name) { &(name), &(name) }`
- `#define LIST_HEAD(name) struct list_head name = LIST_HEAD_INIT(name)`
- `#define INIT_LIST_HEAD(ptr)`
- `#define list_entry(ptr, type, member) ((type *)((char *)(ptr)-(unsigned long)((type *)0->member)))`  
*list\_entry* - get the struct for this entry : the &struct [list\\_head](#) pointer.
- `#define list_for_each(pos, head)`  
*list\_for\_each* - iterate over a list : the &struct [list\\_head](#) to use as a loop counter.
- `#define list_for_each_prev(pos, head)`  
*list\_for\_each\_prev* - iterate over a list backwards : the &struct [list\\_head](#) to use as a loop counter.
- `#define list_for_each_safe(pos, n, head)`  
*list\_for\_each\_safe* - iterate over a list safe against removal of list entry : the &struct [list\\_head](#) to use as a loop counter.
- `#define list_for_each_entry(pos, head, member)`  
*list\_for\_each\_entry* - iterate over list of given type : the type \* to use as a loop counter.
- `#define list_for_each_entry_safe(pos, n, head, member)`  
*list\_for\_each\_entry\_safe* - iterate over list of given type safe against removal of list entry : the type \* to use as a loop counter.
- `#define list_for_each_entry_continue(pos, head, member)`  
*list\_for\_each\_entry\_continue* - iterate over list of given type continuing after existing point : the type \* to use as a loop counter.

## 4.8.1 Macro Definition Documentation

### 4.8.1.1 #define INIT\_LIST\_HEAD( ptr )

**Value:**

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

### 4.8.1.2 #define list\_entry( ptr, type, member )((type \*)((char \*)(ptr)-(unsigned long)&((type \*)0)->member)))

list\_entry - get the struct for this entry : the &struct [list\\_head](#) pointer.

: the type of the struct this is embedded in. : the name of the list\_struct within the struct.

### 4.8.1.3 #define list\_for\_each( pos, head )

**Value:**

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

list\_for\_each - iterate over a list : the &struct [list\\_head](#) to use as a loop counter.

: the head for your list.

### 4.8.1.4 #define list\_for\_each\_entry( pos, head, member )

**Value:**

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

list\_for\_each\_entry - iterate over list of given type : the type \* to use as a loop counter.

: the head for your list. : the name of the list\_struct within the struct.

### 4.8.1.5 #define list\_for\_each\_entry\_continue( pos, head, member )

**Value:**

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

list\_for\_each\_entry\_continue - iterate over list of given type continuing after existing point : the type \* to use as a loop counter.

: the head for your list. : the name of the list\_struct within the struct.

### 4.8.1.6 #define list\_for\_each\_entry\_safe( pos, n, head, member )

**Value:**

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

list\_for\_each\_entry\_safe - iterate over list of given type safe against removal of list entry : the type \* to use as a loop counter.

: another type \* to use as temporary storage : the head for your list. : the name of the list\_struct within the struct.

#### 4.8.1.7 #define list\_for\_each\_prev( pos, head )

**Value:**

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

list\_for\_each\_prev - iterate over a list backwards : the &struct [list\\_head](#) to use as a loop counter.

: the head for your list.

#### 4.8.1.8 #define list\_for\_each\_safe( pos, n, head )

**Value:**

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

list\_for\_each\_safe - iterate over a list safe against removal of list entry : the &struct [list\\_head](#) to use as a loop counter.

: another &struct [list\\_head](#) to use as temporary storage : the head for your list.

#### 4.8.1.9 #define LIST\_HEAD( name ) struct list\_head name = LIST\_HEAD\_INIT(name)

#### 4.8.1.10 #define LIST\_HEAD\_INIT( name ) { &(name), &(name) }

## 4.9 include/hail\_log.h File Reference

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

### Data Structures

- struct [hail\\_log](#)

### Macros

- #define [ATTR\\_PRINTF](#)(x, y)
- #define [HAIL\\_VERBOSE](#)(log,...)  
*Print out a CLD session debug message if enabled.*
- #define [HAIL\\_DEBUG](#)(log,...)  
*Print out an application debug message if enabled.*

- `#define HAIL_INFO(log,...) (log)->func(LOG_INFO, __VA_ARGS__)`  
*Print out an informational log message.*
- `#define HAIL_WARN(log,...) (log)->func(LOG_WARNING, __VA_ARGS__)`  
*Print out a warning message.*
- `#define HAIL_ERR(log,...) (log)->func(LOG_ERR, __VA_ARGS__)`  
*Print out an error message.*
- `#define HAIL_CRIT(log,...) (log)->func(LOG_CRIT, __VA_ARGS__)`  
*Print out a critical warning message.*

#### 4.9.1 Macro Definition Documentation

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

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

Print out a critical warning message.

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

**Value:**

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

Print out an application debug message if enabled.

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

Print out an error message.

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

Print out an informational log message.

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

**Value:**

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

Print out a CLD session debug message if enabled.

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

Print out a warning message.

## 4.10 include/hail\_private.h File Reference

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

## 4.11 include/hstor.h File Reference

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

### Data Structures

- struct [hstor\\_client](#)
- struct [hstor\\_bucket](#)
- struct [hstor\\_blist](#)
- struct [hstor\\_object](#)
- struct [hstor\\_keylist](#)
- struct [http\\_uri](#)
- struct [http\\_hdr](#)
- struct [http\\_req](#)

### Macros

- #define [ARRAY\\_SIZE](#)(arr) (sizeof(arr) / sizeof((arr)[0]))
- #define [PATH\\_ESCAPE\\_MASK](#) 0x02
- #define [QUERY\\_ESCAPE\\_MASK](#) 0x04

### Enumerations

- enum [hstor\\_calling\\_format](#) { [HFMT\\_ORDINARY](#), [HFMT\\_SUBDOMAIN](#) }
- enum { [HREQ\\_MAX\\_HDR](#) = 128 }
- enum [ReqQ](#) { [URIQ\\_ACL](#), [URIQ\\_LOCATION](#), [URIQ\\_LOGGING](#), [URIQ\\_TORRENT](#), [URIQNUM](#) }
- enum [ReqACLC](#) { [ACLC\\_PRIV](#), [ACLC\\_PUB\\_R](#), [ACLC\\_PUB\\_RW](#), [ACLC\\_AUTH\\_R](#), [ACLCNUM](#) }

### Functions

- char \* [hutil\\_time2str](#) (char \*buf, int len, time\_t time)
- time\_t [hutil\\_str2time](#) (const char \*timestr)
- int [hreq\\_hdr\\_push](#) (struct [http\\_req](#) \*req, char \*key, char \*val)
- char \* [hreq\\_hdr](#) (struct [http\\_req](#) \*req, const char \*key)
- void [hreq\\_sign](#) (struct [http\\_req](#) \*req, const char \*bucket, const char \*key, char \*b64hmac\_out)
- GHashTable \* [hreq\\_query](#) (struct [http\\_req](#) \*req)
- int [hreq\\_is\\_query](#) (struct [http\\_req](#) \*req)
- void [hreq\\_free](#) (struct [http\\_req](#) \*req)
- int [hreq\\_acl\\_canned](#) (struct [http\\_req](#) \*req)
- struct [http\\_uri](#) \* [huri\\_parse](#) (struct [http\\_uri](#) \*uri\_dest, char \*uri\_src\_text)
- int [huri\\_field\\_unescape](#) (char \*s, int s\_len)
- char \* [huri\\_field\\_escape](#) (const char \*signed\_str, unsigned char mask)
- void [hstor\\_free](#) (struct [hstor\\_client](#) \*hstor)
- void [hstor\\_free\\_blist](#) (struct [hstor\\_blist](#) \*blist)
- void [hstor\\_free\\_bucket](#) (struct [hstor\\_bucket](#) \*buck)

- void [hstor\\_free\\_object](#) (struct [hstor\\_object](#) \*obj)
- void [hstor\\_free\\_keylist](#) (struct [hstor\\_keylist](#) \*keylist)
- struct [hstor\\_client](#) \* [hstor\\_new](#) (const char \*service\_acc, const char \*service\_host, const char \*user, const char \*secret\_key)
- bool [hstor\\_set\\_format](#) (struct [hstor\\_client](#) \*hstor, enum [hstor\\_calling\\_format](#) f)
- bool [hstor\\_add\\_bucket](#) (struct [hstor\\_client](#) \*hstor, const char \*name)
- bool [hstor\\_del\\_bucket](#) (struct [hstor\\_client](#) \*hstor, const char \*name)
- struct [hstor\\_blist](#) \* [hstor\\_list\\_buckets](#) (struct [hstor\\_client](#) \*hstor)
- bool [hstor\\_get](#) (struct [hstor\\_client](#) \*hstor, const char \*bucket, const char \*key, size\_t(\*write\_cb)(void \*, size\_t, size\_t, void \*), void \*user\_data, bool want\_headers)
- void \* [hstor\\_get\\_inline](#) (struct [hstor\\_client](#) \*hstor, const char \*bucket, const char \*key, bool want\_headers, size\_t \*len)
- bool [hstor\\_put](#) (struct [hstor\\_client](#) \*hstor, const char \*bucket, const char \*key, size\_t(\*read\_cb)(void \*, size\_t, size\_t, void \*), uint64\_t len, void \*user\_data, char \*\*user\_hdrs)
- bool [hstor\\_put\\_inline](#) (struct [hstor\\_client](#) \*hstor, const char \*bucket, const char \*key, void \*data, uint64\_t len, char \*\*user\_hdrs)
- bool [hstor\\_del](#) (struct [hstor\\_client](#) \*hstor, const char \*bucket, const char \*key)
- struct [hstor\\_keylist](#) \* [hstor\\_keys](#) (struct [hstor\\_client](#) \*hstor, const char \*bucket, const char \*prefix, const char \*marker, const char \*delim, unsigned int max\_keys)

#### 4.11.1 Macro Definition Documentation

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

4.11.1.2 `#define PATH_ESCAPE_MASK 0x02`

4.11.1.3 `#define QUERY_ESCAPE_MASK 0x04`

#### 4.11.2 Enumeration Type Documentation

4.11.2.1 anonymous enum

Enumerator

***HREQ\_MAX\_HDR***

4.11.2.2 enum [hstor\\_calling\\_format](#)

Enumerator

***HFMT\_ORDINARY***

***HFMT\_SUBDOMAIN***

4.11.2.3 enum [ReqACLC](#)

Enumerator

***ACLC\_PRIV***

***ACLC\_PUB\_R***

***ACLC\_PUB\_RW***

***ACLC\_AUTH\_R***

***ACLCNUM***

## 4.11.2.4 enum ReqQ

Enumerator

**URIQ\_ACL**  
**URIQ\_LOCATION**  
**URIQ\_LOGGING**  
**URIQ\_TORRENT**  
**URIQNUM**

## 4.11.3 Function Documentation

4.11.3.1 int hreq\_acl\_canned ( struct http\_req \* req )

4.11.3.2 void hreq\_free ( struct http\_req \* req )

4.11.3.3 char\* hreq\_hdr ( struct http\_req \* req, const char \* key )

4.11.3.4 int hreq\_hdr\_push ( struct http\_req \* req, char \* key, char \* val )

4.11.3.5 int hreq\_is\_query ( struct http\_req \* req )

4.11.3.6 GHashTable\* hreq\_query ( struct http\_req \* req )

4.11.3.7 void hreq\_sign ( struct http\_req \* req, const char \* bucket, const char \* key, char \* b64hmac\_out )

4.11.3.8 bool hstor\_add\_bucket ( struct hstor\_client \* hstor, const char \* name )

4.11.3.9 bool hstor\_del ( struct hstor\_client \* hstor, const char \* bucket, const char \* key )

4.11.3.10 bool hstor\_del\_bucket ( struct hstor\_client \* hstor, const char \* name )

4.11.3.11 void hstor\_free ( struct hstor\_client \* hstor )

4.11.3.12 void hstor\_free\_blist ( struct hstor\_blist \* blist )

4.11.3.13 void hstor\_free\_bucket ( struct hstor\_bucket \* buck )

4.11.3.14 void hstor\_free\_keylist ( struct hstor\_keylist \* keylist )

4.11.3.15 void hstor\_free\_object ( struct hstor\_object \* obj )

4.11.3.16 bool hstor\_get ( struct hstor\_client \* hstor, const char \* bucket, const char \* key, size\_t\*(\*)(void \*, size\_t, size\_t, void \*) write\_cb, void \* user\_data, bool want\_headers )

4.11.3.17 void\* hstor\_get\_inline ( struct hstor\_client \* hstor, const char \* bucket, const char \* key, bool want\_headers, size\_t \* len )

4.11.3.18 struct hstor\_keylist\* hstor\_keys ( struct hstor\_client \* hstor, const char \* bucket, const char \* prefix, const char \* marker, const char \* delim, unsigned int max\_keys )

4.11.3.19 struct hstor\_blist\* hstor\_list\_buckets ( struct hstor\_client \* hstor )

4.11.3.20 struct hstor\_client\* hstor\_new ( const char \* service\_acc, const char \* service\_host, const char \* user, const char \* secret\_key )



- 4.11.3.21 `bool hstor_put ( struct hstor_client * hstor, const char * bucket, const char * key, size_t (void *, size_t, size_t, void *) read_cb, uint64_t len, void * user_data, char ** user_hdrs )`
- 4.11.3.22 `bool hstor_put_inline ( struct hstor_client * hstor, const char * bucket, const char * key, void * data, uint64_t len, char ** user_hdrs )`
- 4.11.3.23 `bool hstor_set_format ( struct hstor_client * hstor, enum hstor_calling_format f )`
- 4.11.3.24 `char* huri_field_escape ( const char * signed_str, unsigned char mask )`
- 4.11.3.25 `int huri_field_unescape ( char * s, int s_len )`
- 4.11.3.26 `struct http_uri* huri_parse ( struct http_uri * uri_dest, char * uri_src_text )`
- 4.11.3.27 `time_t hutil_str2time ( const char * timestr )`
- 4.11.3.28 `char* hutil_time2str ( char * buf, int len, time_t time )`

## 4.12 include/ncld.h File Reference

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

### Data Structures

- struct [ncld\\_sess](#)
- struct [ncld\\_fh](#)
- struct [ncld\\_read](#)

### Functions

- struct [ncld\\_sess](#) \* [ncld\\_sess\\_open](#) (const char \*host, int port, int \*error, void(\*event)(void \*, unsigned int), void \*ev\_arg, const char \*cld\_user, const char \*cld\_key, struct [hail\\_log](#) \*log)
- struct [ncld\\_fh](#) \* [ncld\\_open](#) (struct [ncld\\_sess](#) \*s, const char \*fname, unsigned int mode, int \*error, unsigned int events, void(\*event)(void \*, unsigned int), void \*ev\_arg)
- int [ncld\\_del](#) (struct [ncld\\_sess](#) \*nsess, const char \*fname)
- struct [ncld\\_read](#) \* [ncld\\_get](#) (struct [ncld\\_fh](#) \*fh, int \*error)
- struct [ncld\\_read](#) \* [ncld\\_get\\_meta](#) (struct [ncld\\_fh](#) \*fh, int \*error)
- void [ncld\\_read\\_free](#) (struct [ncld\\_read](#) \*rp)
- int [ncld\\_write](#) (struct [ncld\\_fh](#) \*, const void \*data, long len)
- int [ncld\\_trylock](#) (struct [ncld\\_fh](#) \*)
- int [ncld\\_qlock](#) (struct [ncld\\_fh](#) \*)
- int [ncld\\_unlock](#) (struct [ncld\\_fh](#) \*)
- void [ncld\\_close](#) (struct [ncld\\_fh](#) \*)
- void [ncld\\_sess\\_close](#) (struct [ncld\\_sess](#) \*s)
- void [ncld\\_init](#) (void)

### 4.12.1 Function Documentation

- 4.12.1.1 `void ncld_close ( struct ncld_fh * )`

- 4.12.1.2 `int nclد_del ( struct nclد_sess * nsess, const char * fname )`
- 4.12.1.3 `struct nclد_read* nclد_get ( struct nclد_fh * fh, int * error )`
- 4.12.1.4 `struct nclد_read* nclد_get_meta ( struct nclد_fh * fh, int * error )`
- 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 )`
- 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 )`
- 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](#)

### Macros

- #define [OC\\_F\\_DIRTY](#) 0x1
- #define [objcache\\_get](#)(c, k, l) [\\_\\_objcache\\_get](#)(c, k, l, 0)
- #define [objcache\\_get\\_dirty](#)(c, k, l) [\\_\\_objcache\\_get](#)(c, k, l, [OC\\_F\\_DIRTY](#))

### Functions

- struct [objcache\\_entry](#) \* [\\_\\_objcache\\_get](#) (struct [objcache](#) \*cache, const char \*key, int klen, unsigned int flag)
- bool [objcache\\_test\\_dirty](#) (struct [objcache](#) \*cache, struct [objcache\\_entry](#) \*entry)
- void [objcache\\_put](#) (struct [objcache](#) \*cache, struct [objcache\\_entry](#) \*entry)
- int [objcache\\_count](#) (struct [objcache](#) \*cache)
- int [objcache\\_init](#) (struct [objcache](#) \*cache)
- void [objcache\\_fini](#) (struct [objcache](#) \*cache)

### 4.13.1 Macro Definition Documentation

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

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

4.13.1.3 `#define OC_F_DIRTY 0x1`

### 4.13.2 Function Documentation

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

4.13.2.2 `int objcache_count ( struct objcache * cache )`

4.13.2.3 `void objcache_fini ( struct objcache * cache )`

4.13.2.4 `int objcache_init ( struct objcache * cache )`

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

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

# Index

ACLC\_AUTH\_R  
    hstor.h, [41](#)

ACLC\_PRIV  
    hstor.h, [41](#)

ACLC\_PUB\_R  
    hstor.h, [41](#)

ACLC\_PUB\_RW  
    hstor.h, [41](#)

ACLCNUM  
    hstor.h, [41](#)

CHD\_CSUM\_SZ  
    chunk\_msg.h, [28](#)

CHD\_KEY\_SZ  
    chunk\_msg.h, [28](#)

CHD\_MAGIC\_SZ  
    chunk\_msg.h, [28](#)

CHD\_SIG\_SZ  
    chunk\_msg.h, [28](#)

CHD\_USER\_SZ  
    chunk\_msg.h, [28](#)

CHF\_SYNC  
    chunk\_msg.h, [29](#)

CHF\_TBL\_CREAT  
    chunk\_msg.h, [29](#)

CHF\_TBL\_EXCL  
    chunk\_msg.h, [29](#)

CHO\_CHECK\_START  
    chunk\_msg.h, [29](#)

CHO\_CHECK\_STATUS  
    chunk\_msg.h, [29](#)

CHO\_CP  
    chunk\_msg.h, [29](#)

CHO\_DEL  
    chunk\_msg.h, [29](#)

CHO\_GET  
    chunk\_msg.h, [29](#)

CHO\_GET\_META  
    chunk\_msg.h, [29](#)

CHO\_LIST  
    chunk\_msg.h, [29](#)

CHO\_LOGIN  
    chunk\_msg.h, [29](#)

CHO\_NOP  
    chunk\_msg.h, [29](#)

CHO\_PUT  
    chunk\_msg.h, [29](#)

CHO\_START\_TLS  
    chunk\_msg.h, [29](#)

CHO\_TABLE\_OPEN  
    chunk\_msg.h, [29](#)

che\_AccessDenied  
    chunk\_msg.h, [28](#)

che\_Busy  
    chunk\_msg.h, [29](#)

che\_InternalError  
    chunk\_msg.h, [28](#)

che\_InvalidArgument  
    chunk\_msg.h, [28](#)

che\_InvalidKey  
    chunk\_msg.h, [28](#)

che\_InvalidTable  
    chunk\_msg.h, [28](#)

che\_InvalidURI  
    chunk\_msg.h, [28](#)

che\_KeyExists  
    chunk\_msg.h, [29](#)

che\_NoSuchKey  
    chunk\_msg.h, [28](#)

che\_SignatureDoesNotMatch  
    chunk\_msg.h, [28](#)

che\_Success  
    chunk\_msg.h, [28](#)

chk\_Active  
    chunk\_msg.h, [28](#)

chk\_Idle  
    chunk\_msg.h, [28](#)

chk\_Off  
    chunk\_msg.h, [28](#)

chunk\_msg.h  
    CHD\_CSUM\_SZ, [28](#)  
    CHD\_KEY\_SZ, [28](#)  
    CHD\_MAGIC\_SZ, [28](#)  
    CHD\_SIG\_SZ, [28](#)  
    CHD\_USER\_SZ, [28](#)  
    CHF\_SYNC, [29](#)  
    CHF\_TBL\_CREAT, [29](#)  
    CHF\_TBL\_EXCL, [29](#)  
    CHO\_CHECK\_START, [29](#)  
    CHO\_CHECK\_STATUS, [29](#)  
    CHO\_CP, [29](#)  
    CHO\_DEL, [29](#)  
    CHO\_GET, [29](#)  
    CHO\_GET\_META, [29](#)  
    CHO\_LIST, [29](#)  
    CHO\_LOGIN, [29](#)  
    CHO\_NOP, [29](#)  
    CHO\_PUT, [29](#)  
    CHO\_START\_TLS, [29](#)

- CHO\_TABLE\_OPEN, [29](#)
- che\_AccessDenied, [28](#)
- che\_Busy, [29](#)
- che\_InternalError, [28](#)
- che\_InvalidArgument, [28](#)
- che\_InvalidKey, [28](#)
- che\_InvalidTable, [28](#)
- che\_InvalidURI, [28](#)
- che\_KeyExists, [29](#)
- che\_NoSuchKey, [28](#)
- che\_SignatureDoesNotMatch, [28](#)
- che\_Success, [28](#)
- chk\_Active, [28](#)
- chk\_Idle, [28](#)
- chk\_Off, [28](#)

HFMT\_ORDINARY

- hstor.h, [41](#)

HFMT\_SUBDOMAIN

- hstor.h, [41](#)

HREQ\_MAX\_HDR

- hstor.h, [41](#)

hstor.h

- ACLC\_AUTH\_R, [41](#)
- ACLC\_PRIV, [41](#)
- ACLC\_PUB\_R, [41](#)
- ACLC\_PUB\_RW, [41](#)
- ACLCNUM, [41](#)
- HFMT\_ORDINARY, [41](#)
- HFMT\_SUBDOMAIN, [41](#)
- HREQ\_MAX\_HDR, [41](#)
- URIQ\_ACL, [42](#)
- URIQ\_LOCATION, [42](#)
- URIQ\_LOGGING, [42](#)
- URIQ\_TORRENT, [42](#)
- URIQNUM, [42](#)

lock

- objcache, [24](#)

objcache, [23](#)

- lock, [24](#)
- table, [24](#)

table

- objcache, [24](#)

URIQ\_ACL

- hstor.h, [42](#)

URIQ\_LOCATION

- hstor.h, [42](#)

URIQ\_LOGGING

- hstor.h, [42](#)

URIQ\_TORRENT

- hstor.h, [42](#)

URIQNUM

- hstor.h, [42](#)