

CLD

0.1git

Generated by Doxygen 1.7.5

Thu Oct 6 2011 13:18:13



# 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 . . . . .	7
3.3.1.1	data_len . . . . .	7
3.3.1.2	hash . . . . .	7

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

3.10 cldc_fh Struct Reference . . . . .	10
3.10.1 Detailed Description . . . . .	10
3.10.2 Field Documentation . . . . .	11
3.10.2.1 fh . . . . .	11
3.10.2.2 sess . . . . .	11
3.10.2.3 valid . . . . .	11
3.11 cldc_host Struct Reference . . . . .	11
3.11.1 Detailed Description . . . . .	11
3.11.2 Field Documentation . . . . .	11
3.11.2.1 host . . . . .	11
3.11.2.2 port . . . . .	11
3.11.2.3 prio . . . . .	11
3.11.2.4 weight . . . . .	11
3.12 cldc_msg Struct Reference . . . . .	12
3.12.1 Detailed Description . . . . .	12
3.12.2 Field Documentation . . . . .	12
3.12.2.1 cb . . . . .	12
3.12.2.2 cb_private . . . . .	12
3.12.2.3 copts . . . . .	12
3.12.2.4 done . . . . .	12
3.12.2.5 expire_time . . . . .	12
3.12.2.6 n_pkts . . . . .	12
3.12.2.7 op . . . . .	12
3.12.2.8 pkt_info . . . . .	12
3.12.2.9 sess . . . . .	12
3.12.2.10 xid . . . . .	13
3.13 cldc_node_metadata Struct Reference . . . . .	13
3.13.1 Field Documentation . . . . .	13
3.13.1.1 flags . . . . .	13
3.13.1.2 inode_name . . . . .	13
3.13.1.3 inum . . . . .	13
3.13.1.4 time_create . . . . .	13
3.13.1.5 time_modify . . . . .	13
3.13.1.6 vers . . . . .	13

3.14 cldc_ops Struct Reference . . . . .	13
3.14.1 Detailed Description . . . . .	14
3.14.2 Field Documentation . . . . .	14
3.14.2.1 event . . . . .	14
3.14.2.2 pkt_send . . . . .	14
3.14.2.3 timer_ctl . . . . .	14
3.15 cldc_pkt_info Struct Reference . . . . .	14
3.15.1 Field Documentation . . . . .	14
3.15.1.1 data . . . . .	14
3.15.1.2 hdr_len . . . . .	15
3.15.1.3 pkt_len . . . . .	15
3.15.1.4 retries . . . . .	15
3.15.1.5 user . . . . .	15
3.16 cldc_session Struct Reference . . . . .	15
3.16.1 Detailed Description . . . . .	15
3.16.2 Field Documentation . . . . .	16
3.16.2.1 addr . . . . .	16
3.16.2.2 addr_len . . . . .	16
3.16.2.3 cfh . . . . .	16
3.16.2.4 confirmed . . . . .	16
3.16.2.5 expire_time . . . . .	16
3.16.2.6 expired . . . . .	16
3.16.2.7 inode_name_temp . . . . .	16
3.16.2.8 log . . . . .	16
3.16.2.9 msg_buf . . . . .	16
3.16.2.10 msg_buf_len . . . . .	16
3.16.2.11 msg_buf_op . . . . .	16
3.16.2.12 msg_scan_time . . . . .	16
3.16.2.13 next_seqid_in . . . . .	16
3.16.2.14 next_seqid_in_tr . . . . .	16
3.16.2.15 next_seqid_out . . . . .	16
3.16.2.16 ops . . . . .	16
3.16.2.17 out_msg . . . . .	16
3.16.2.18 payload . . . . .	16

---

3.16.2.19 private . . . . .	16
3.16.2.20 secret_key . . . . .	16
3.16.2.21 sid . . . . .	16
3.16.2.22 user . . . . .	16
3.17 cldc_udp Struct Reference . . . . .	17
3.17.1 Detailed Description . . . . .	17
3.17.2 Field Documentation . . . . .	17
3.17.2.1 addr . . . . .	17
3.17.2.2 addr_len . . . . .	17
3.17.2.3 cb . . . . .	17
3.17.2.4 cb_private . . . . .	17
3.17.2.5 fd . . . . .	17
3.17.2.6 sess . . . . .	17
3.18 hail_log Struct Reference . . . . .	17
3.18.1 Field Documentation . . . . .	18
3.18.1.1 debug . . . . .	18
3.18.1.2 func . . . . .	18
3.18.1.3 verbose . . . . .	18
3.19 hstor_blist Struct Reference . . . . .	18
3.19.1 Field Documentation . . . . .	18
3.19.1.1 list . . . . .	18
3.19.1.2 own_id . . . . .	18
3.19.1.3 own_name . . . . .	18
3.20 hstor_bucket Struct Reference . . . . .	18
3.20.1 Field Documentation . . . . .	19
3.20.1.1 name . . . . .	19
3.20.1.2 time_create . . . . .	19
3.21 hstor_client Struct Reference . . . . .	19
3.21.1 Field Documentation . . . . .	19
3.21.1.1 acc . . . . .	19
3.21.1.2 curl . . . . .	19
3.21.1.3 host . . . . .	19
3.21.1.4 key . . . . .	19
3.21.1.5 user . . . . .	19

---

3.21.1.6	verbose	19
3.22	hstor_keylist Struct Reference	20
3.22.1	Field Documentation	20
3.22.1.1	common_pfx	20
3.22.1.2	contents	20
3.22.1.3	delim	20
3.22.1.4	marker	20
3.22.1.5	max_keys	20
3.22.1.6	name	20
3.22.1.7	prefix	20
3.22.1.8	trunc	20
3.23	hstor_object Struct Reference	20
3.23.1	Field Documentation	21
3.23.1.1	etag	21
3.23.1.2	key	21
3.23.1.3	own_id	21
3.23.1.4	own_name	21
3.23.1.5	size	21
3.23.1.6	storage	21
3.23.1.7	time_mod	21
3.24	http_hdr Struct Reference	21
3.24.1	Field Documentation	21
3.24.1.1	key	21
3.24.1.2	val	22
3.25	http_req Struct Reference	22
3.25.1	Field Documentation	22
3.25.1.1	hdr	22
3.25.1.2	major	22
3.25.1.3	method	22
3.25.1.4	minor	22
3.25.1.5	n_hdr	22
3.25.1.6	orig_path	22
3.25.1.7	uri	22
3.26	http_uri Struct Reference	22



3.26.1	Field Documentation	23
3.26.1.1	fragment	23
3.26.1.2	fragment_len	23
3.26.1.3	hostname	23
3.26.1.4	hostname_len	23
3.26.1.5	path	23
3.26.1.6	path_len	23
3.26.1.7	port	23
3.26.1.8	query	23
3.26.1.9	query_len	23
3.26.1.10	scheme	23
3.26.1.11	scheme_len	23
3.26.1.12	userinfo	23
3.26.1.13	userinfo_len	23
3.27	list_head Struct Reference	24
3.27.1	Field Documentation	24
3.27.1.1	next	24
3.27.1.2	prev	24
3.28	nclد_fh Struct Reference	24
3.28.1	Field Documentation	24
3.28.1.1	errc	24
3.28.1.2	event_arg	24
3.28.1.3	event_func	24
3.28.1.4	event_mask	25
3.28.1.5	fh	25
3.28.1.6	is_open	25
3.28.1.7	nios	25
3.28.1.8	sess	25
3.29	nclد_read Struct Reference	25
3.29.1	Field Documentation	25
3.29.1.1	errc	25
3.29.1.2	fh	25
3.29.1.3	is_done	25
3.29.1.4	length	25

3.29.1.5	meta	25
3.29.1.6	ptr	25
3.30	ncld_sess Struct Reference	26
3.30.1	Field Documentation	26
3.30.1.1	cond	26
3.30.1.2	errc	26
3.30.1.3	event	26
3.30.1.4	event_arg	26
3.30.1.5	handles	26
3.30.1.6	host	26
3.30.1.7	is_up	26
3.30.1.8	mutex	26
3.30.1.9	open_done	26
3.30.1.10	port	26
3.30.1.11	thread	26
3.30.1.12	tlist	27
3.30.1.13	to_thread	27
3.30.1.14	udp	27
3.30.1.15	udp_timer	27
3.31	objcache Struct Reference	27
3.31.1	Field Documentation	27
3.31.1.1	lock	27
3.31.1.2	table	27
3.32	objcache_entry Struct Reference	27
3.32.1	Field Documentation	28
3.32.1.1	flags	28
3.32.1.2	hash	28
3.32.1.3	ref	28
3.33	st_client Struct Reference	28
3.33.1	Field Documentation	28
3.33.1.1	fd	28
3.33.1.2	host	28
3.33.1.3	key	28
3.33.1.4	req_buf	28

3.33.1.5	ssl	28
3.33.1.6	ssl_ctx	28
3.33.1.7	user	28
3.33.1.8	verbose	28
3.34	st_keylist Struct Reference	29
3.34.1	Field Documentation	29
3.34.1.1	contents	29
3.34.1.2	name	29
3.35	st_object Struct Reference	29
3.35.1	Field Documentation	29
3.35.1.1	etag	29
3.35.1.2	name	29
3.35.1.3	owner	29
3.35.1.4	size	30
3.35.1.5	time_mod	30
<b>4</b>	<b>File Documentation</b>	<b>31</b>
4.1	include/chunk-private.h File Reference	31
4.1.1	Define Documentation	31
4.1.1.1	BAD_TPATH_FMT	31
4.1.1.2	MDB_TPATH_FMT	31
4.1.1.3	PREFIX_LEN	31
4.2	include/chunk_msg.h File Reference	31
4.2.1	Define Documentation	32
4.2.1.1	CHUNKD_MAGIC	32
4.2.2	Enumeration Type Documentation	32
4.2.2.1	anonymous enum	32
4.2.2.2	chunk_check_state	32
4.2.2.3	chunk_errcode	33
4.2.2.4	chunk_flags	33
4.2.2.5	chunksrv_ops	33
4.3	include/chunkc.h File Reference	34
4.3.1	Function Documentation	35
4.3.1.1	stc_check_start	35

4.3.1.2	<a href="#">stc_check_status</a>	35
4.3.1.3	<a href="#">stc_cp</a>	35
4.3.1.4	<a href="#">stc_del</a>	35
4.3.1.5	<a href="#">stc_free</a>	35
4.3.1.6	<a href="#">stc_free_keylist</a>	35
4.3.1.7	<a href="#">stc_free_object</a>	35
4.3.1.8	<a href="#">stc_get</a>	35
4.3.1.9	<a href="#">stc_get_inline</a>	35
4.3.1.10	<a href="#">stc_get_recv</a>	35
4.3.1.11	<a href="#">stc_get_start</a>	35
4.3.1.12	<a href="#">stc_init</a>	35
4.3.1.13	<a href="#">stc_keys</a>	35
4.3.1.14	<a href="#">stc_new</a>	35
4.3.1.15	<a href="#">stc_ping</a>	35
4.3.1.16	<a href="#">stc_put</a>	35
4.3.1.17	<a href="#">stc_put_inline</a>	35
4.3.1.18	<a href="#">stc_put_send</a>	35
4.3.1.19	<a href="#">stc_put_start</a>	35
4.3.1.20	<a href="#">stc_put_sync</a>	36
4.3.1.21	<a href="#">stc_readport</a>	36
4.3.1.22	<a href="#">stc_table_open</a>	36
4.4	<a href="#">include/chunksrv.h File Reference</a>	36
4.4.1	<a href="#">Function Documentation</a>	36
4.4.1.1	<a href="#">chreq_sign</a>	36
4.4.1.2	<a href="#">req_len</a>	36
4.5	<a href="#">include/cld-private.h File Reference</a>	36
4.6	<a href="#">include/cld_common.h File Reference</a>	36
4.6.1	<a href="#">Define Documentation</a>	37
4.6.1.1	<a href="#">CLD_ALIGN8</a>	37
4.6.1.2	<a href="#">CLD_PKT_FTR_LEN</a>	37
4.6.1.3	<a href="#">PKT_HDR_TO_STR_SCRATCH_LEN</a>	37
4.6.1.4	<a href="#">SIDARG</a>	37
4.6.1.5	<a href="#">SIDFMT</a>	37
4.6.2	<a href="#">Function Documentation</a>	37

4.6.2.1	<a href="#">__attribute__</a>	37
4.6.2.2	<a href="#">__cld_dump_buf</a>	38
4.6.2.3	<a href="#">cld_authcheck</a>	38
4.6.2.4	<a href="#">cld_authsign</a>	38
4.6.2.5	<a href="#">cld_errstr</a>	38
4.6.2.6	<a href="#">cld_opstr</a>	38
4.6.2.7	<a href="#">cld_pkt_hdr_to_str</a>	38
4.6.2.8	<a href="#">cld_rand64</a>	38
4.6.2.9	<a href="#">cld_readport</a>	38
4.6.2.10	<a href="#">cld_sid2llu</a>	38
4.6.2.11	<a href="#">cld_timer_add</a>	38
4.6.2.12	<a href="#">cld_timer_del</a>	38
4.6.2.13	<a href="#">cld_timers_run</a>	38
4.7	<a href="#">include/cldc.h File Reference</a>	38
4.7.1	<a href="#">Function Documentation</a>	40
4.7.1.1	<a href="#">cldc_close</a>	40
4.7.1.2	<a href="#">cldc_copts_get_data</a>	40
4.7.1.3	<a href="#">cldc_copts_get_metadata</a>	40
4.7.1.4	<a href="#">cldc_del</a>	40
4.7.1.5	<a href="#">cldc_dirent_count</a>	40
4.7.1.6	<a href="#">cldc_dirent_cur_fini</a>	40
4.7.1.7	<a href="#">cldc_dirent_cur_init</a>	40
4.7.1.8	<a href="#">cldc_dirent_first</a>	40
4.7.1.9	<a href="#">cldc_dirent_name</a>	40
4.7.1.10	<a href="#">cldc_dirent_next</a>	40
4.7.1.11	<a href="#">cldc_end_sess</a>	40
4.7.1.12	<a href="#">cldc_get</a>	40
4.7.1.13	<a href="#">cldc_getaddr</a>	40
4.7.1.14	<a href="#">cldc_init</a>	40
4.7.1.15	<a href="#">cldc_kill_sess</a>	40
4.7.1.16	<a href="#">cldc_lock</a>	41
4.7.1.17	<a href="#">cldc_new_sess</a>	41
4.7.1.18	<a href="#">cldc_nop</a>	41
4.7.1.19	<a href="#">cldc_open</a>	41

4.7.1.20	cldc_put	41
4.7.1.21	cldc_receive_pkt	41
4.7.1.22	cldc_saveaddr	41
4.7.1.23	cldc_udp_free	41
4.7.1.24	cldc_udp_new	41
4.7.1.25	cldc_udp_pkt_send	41
4.7.1.26	cldc_udp_receive_pkt	41
4.7.1.27	cldc_unlock	41
4.8	include/elist.h File Reference	42
4.8.1	Define Documentation	42
4.8.1.1	INIT_LIST_HEAD	42
4.8.1.2	list_entry	43
4.8.1.3	list_for_each	43
4.8.1.4	list_for_each_entry	43
4.8.1.5	list_for_each_entry_continue	43
4.8.1.6	list_for_each_entry_safe	43
4.8.1.7	list_for_each_prev	44
4.8.1.8	list_for_each_safe	44
4.8.1.9	LIST_HEAD	44
4.8.1.10	LIST_HEAD_INIT	44
4.9	include/hail_log.h File Reference	44
4.9.1	Define Documentation	45
4.9.1.1	ATTR_PRINTF	45
4.9.1.2	HAIL_CRIT	45
4.9.1.3	HAIL_DEBUG	45
4.9.1.4	HAIL_ERR	45
4.9.1.5	HAIL_INFO	45
4.9.1.6	HAIL_VERBOSE	46
4.9.1.7	HAIL_WARN	46
4.10	include/hail_private.h File Reference	46
4.10.1	Function Documentation	46
4.10.1.1	xdr_sizeof	46
4.11	include/hstor.h File Reference	46
4.11.1	Define Documentation	48

4.11.1.1	ARRAY_SIZE	48
4.11.1.2	PATH_ESCAPE_MASK	48
4.11.1.3	QUERY_ESCAPE_MASK	48
4.11.2	Enumeration Type Documentation	48
4.11.2.1	anonymous enum	48
4.11.2.2	ReqACLC	48
4.11.2.3	ReqQ	48
4.11.3	Function Documentation	49
4.11.3.1	hreq_acl_canned	49
4.11.3.2	hreq_free	49
4.11.3.3	hreq_hdr	49
4.11.3.4	hreq_hdr_push	49
4.11.3.5	hreq_is_query	49
4.11.3.6	hreq_query	49
4.11.3.7	hreq_sign	49
4.11.3.8	hstor_add_bucket	49
4.11.3.9	hstor_del	49
4.11.3.10	hstor_del_bucket	49
4.11.3.11	hstor_free	49
4.11.3.12	hstor_free_blist	49
4.11.3.13	hstor_free_bucket	49
4.11.3.14	hstor_free_keylist	49
4.11.3.15	hstor_free_object	49
4.11.3.16	hstor_get	49
4.11.3.17	hstor_get_inline	49
4.11.3.18	hstor_keys	49
4.11.3.19	hstor_list_buckets	49
4.11.3.20	hstor_new	49
4.11.3.21	hstor_put	50
4.11.3.22	hstor_put_inline	50
4.11.3.23	huri_field_escape	50
4.11.3.24	huri_field_unescape	50
4.11.3.25	huri_parse	50
4.11.3.26	hutil_str2time	50

4.11.3.27	hutil_time2str . . . . .	50
4.12	include/ncl.d.h File Reference . . . . .	50
4.12.1	Function Documentation . . . . .	51
4.12.1.1	ncl.d_close . . . . .	51
4.12.1.2	ncl.d_del . . . . .	51
4.12.1.3	ncl.d_get . . . . .	51
4.12.1.4	ncl.d_get_meta . . . . .	51
4.12.1.5	ncl.d_init . . . . .	51
4.12.1.6	ncl.d_open . . . . .	51
4.12.1.7	ncl.d_qlock . . . . .	51
4.12.1.8	ncl.d_read_free . . . . .	51
4.12.1.9	ncl.d_sess_close . . . . .	51
4.12.1.10	ncl.d_sess_open . . . . .	51
4.12.1.11	ncl.d_trylock . . . . .	51
4.12.1.12	ncl.d_unlock . . . . .	51
4.12.1.13	ncl.d_write . . . . .	51
4.13	include/objcache.h File Reference . . . . .	51
4.13.1	Define Documentation . . . . .	52
4.13.1.1	objcache_get . . . . .	52
4.13.1.2	objcache_get_dirty . . . . .	52
4.13.1.3	OC_F_DIRTY . . . . .	52
4.13.2	Function Documentation . . . . .	52
4.13.2.1	__objcache_get . . . . .	52
4.13.2.2	objcache_count . . . . .	52
4.13.2.3	objcache_fini . . . . .	52
4.13.2.4	objcache_init . . . . .	52
4.13.2.5	objcache_put . . . . .	52
4.13.2.6	objcache_test_dirty . . . . .	52



# 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>	8
<a href="#">cld_timer</a>	8
<a href="#">cld_timer_list</a>	9
<a href="#">cldc_call_opts</a>	
Per-operation application options	10
<a href="#">cldc_fh</a>	
Open file handle associated with a session	10
<a href="#">cldc_host</a>	
Information for a single CLD server host	11
<a href="#">cldc_msg</a>	
Outgoing message, from client to server	12
<a href="#">cldc_node_metadata</a>	13
<a href="#">cldc_ops</a>	
Application-supplied facilities	13
<a href="#">cldc_pkt_info</a>	14
<a href="#">cldc_session</a>	
Single CLD client session	15
<a href="#">cldc_udp</a>	
A UDP implementation of the CLD client protocol	17
<a href="#">hail_log</a>	17
<a href="#">hstor_blist</a>	18
<a href="#">hstor_bucket</a>	18
<a href="#">hstor_client</a>	19
<a href="#">hstor_keylist</a>	20

hstor_object . . . . .	20
http_hdr . . . . .	21
http_req . . . . .	22
http_uri . . . . .	22
list_head . . . . .	24
nclد_fh . . . . .	24
nclد_read . . . . .	25
nclد_sess . . . . .	26
objcache . . . . .	27
objcache_entry . . . . .	27
st_client . . . . .	28
st_keylist . . . . .	29
st_object . . . . .	29

## 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> . . . . .	31
include/ <a href="#">chunk_msg.h</a> . . . . .	31
include/ <a href="#">chunkc.h</a> . . . . .	34
include/ <a href="#">chunksrv.h</a> . . . . .	36
include/ <a href="#">cld-private.h</a> . . . . .	36
include/ <a href="#">cld_common.h</a> . . . . .	36
include/ <a href="#">cldc.h</a> . . . . .	38
include/ <a href="#">elist.h</a> . . . . .	42
include/ <a href="#">hail_log.h</a> . . . . .	44
include/ <a href="#">hail_private.h</a> . . . . .	46
include/ <a href="#">hstor.h</a> . . . . .	46
include/ <a href="#">ncld.h</a> . . . . .	50
include/ <a href="#">objcache.h</a> . . . . .	51



## 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]
- struct cldc\_ops \* ops
- struct hail\_log log
- void \* private
- uint8\_t addr [64]
- size\_t addr\_len
- GList \* cfh
- GList \* out\_msg
- time\_t msg\_scan\_time
- time\_t expire\_time
- bool expired
- uint64\_t next\_seqid\_in
- uint64\_t next\_seqid\_in\_tr
- uint64\_t next\_seqid\_out
- char user [CLD\_MAX\_USERNAME]
- char secret\_key [CLD\_MAX\_SECRET\_KEY]
- bool confirmed
- enum cld\_msg\_op msg\_buf\_op
- unsigned int msg\_buf\_len
- char msg\_buf [CLD\_MAX\_MSG\_SZ]
- char payload [CLD\_MAX\_PAYLOAD\_SZ]
- char inode\_name\_temp [CLD\_INODE\_NAME\_MAX]

### 3.16.1 Detailed Description

a single CLD client session

### 3.16.2 Field Documentation

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

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



- include/[cldc.h](#)

## 3.17 cldc\_udp Struct Reference

A UDP implementation of the CLD client protocol.

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

### Data Fields

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

### 3.17.1 Detailed Description

A UDP implementation of the CLD client protocol.

### 3.17.2 Field Documentation

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

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

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

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

3.17.2.5 int cldc\_udp::fd

3.17.2.6 struct cldc\_session\* cldc\_udp::sess

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

- include/[cldc.h](#)

## 3.18 hail\_log Struct Reference

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

## Data Fields

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

### 3.18.1 Field Documentation

3.18.1.1 void(\*) boo [hail\\_log::debug](#))

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

3.18.1.3 bool [hail\\_log::verbose](#)

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

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

## 3.19 hstor\_blist Struct Reference

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

## Data Fields

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

### 3.19.1 Field Documentation

3.19.1.1 GList\* [hstor\\_blist::list](#)

3.19.1.2 char\* [hstor\\_blist::own\\_id](#)

3.19.1.3 char\* [hstor\\_blist::own\\_name](#)

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

- include/[hstor.h](#)

## 3.20 hstor\_bucket Struct Reference

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

### Data Fields

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

#### 3.20.1 Field Documentation

3.20.1.1 char\* hstor\_bucket::name

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

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

- include/[hstor.h](#)

## 3.21 hstor\_client Struct Reference

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

### Data Fields

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

#### 3.21.1 Field Documentation

3.21.1.1 char\* hstor\_client::acc

3.21.1.2 CURL\* hstor\_client::curl

3.21.1.3 char\* hstor\_client::host

3.21.1.4 char\* hstor\_client::key

3.21.1.5 char\* hstor\_client::user

3.21.1.6 bool hstor\_client::verbose

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

- include/[hstor.h](#)

## 3.22 hstor\_keylist Struct Reference

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

### Data Fields

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

### 3.22.1 Field Documentation

3.22.1.1 GList\* [hstor\\_keylist::common\\_pfx](#)

3.22.1.2 GList\* [hstor\\_keylist::contents](#)

3.22.1.3 char\* [hstor\\_keylist::delim](#)

3.22.1.4 char\* [hstor\\_keylist::marker](#)

3.22.1.5 unsigned int [hstor\\_keylist::max\\_keys](#)

3.22.1.6 char\* [hstor\\_keylist::name](#)

3.22.1.7 char\* [hstor\\_keylist::prefix](#)

3.22.1.8 bool [hstor\\_keylist::trunc](#)

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

- include/[hstor.h](#)

## 3.23 hstor\_object Struct Reference

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

### Data Fields

- char \* [key](#)
- char \* [time\\_mod](#)
- char \* [etag](#)
- uint64\_t [size](#)
- char \* [storage](#)
- char \* [own\\_id](#)
- char \* [own\\_name](#)

#### 3.23.1 Field Documentation

3.23.1.1 char\* hstor\_object::etag

3.23.1.2 char\* hstor\_object::key

3.23.1.3 char\* hstor\_object::own\_id

3.23.1.4 char\* hstor\_object::own\_name

3.23.1.5 uint64\_t hstor\_object::size

3.23.1.6 char\* hstor\_object::storage

3.23.1.7 char\* hstor\_object::time\_mod

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

- include/[hstor.h](#)

## 3.24 http\_hdr Struct Reference

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

### Data Fields

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

#### 3.24.1 Field Documentation

3.24.1.1 char\* http\_hdr::key

### 3.24.1.2 char\* http\_hdr::val

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

- include/[hstor.h](#)

## 3.25 http\_req Struct Reference

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

### Data Fields

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

### 3.25.1 Field Documentation

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

3.25.1.2 int [http\\_req::major](#)

3.25.1.3 char\* [http\\_req::method](#)

3.25.1.4 int [http\\_req::minor](#)

3.25.1.5 unsigned int [http\\_req::n\\_hdr](#)

3.25.1.6 char\* [http\\_req::orig\\_path](#)

3.25.1.7 struct [http\\_uri](#) [http\\_req::uri](#)

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

- include/[hstor.h](#)

## 3.26 http\_uri Struct Reference

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

## Data Fields

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

### 3.26.1 Field Documentation

3.26.1.1 char\* `http_uri::fragment`

3.26.1.2 unsigned int `http_uri::fragment_len`

3.26.1.3 char\* `http_uri::hostname`

3.26.1.4 unsigned int `http_uri::hostname_len`

3.26.1.5 char\* `http_uri::path`

3.26.1.6 unsigned int `http_uri::path_len`

3.26.1.7 unsigned int `http_uri::port`

3.26.1.8 char\* `http_uri::query`

3.26.1.9 unsigned int `http_uri::query_len`

3.26.1.10 char\* `http_uri::scheme`

3.26.1.11 unsigned int `http_uri::scheme_len`

3.26.1.12 char\* `http_uri::userinfo`

3.26.1.13 unsigned int `http_uri::userinfo_len`

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

- [include/hstor.h](#)

## 3.27 list\_head Struct Reference

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

### Data Fields

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

### 3.27.1 Field Documentation

3.27.1.1 struct [list\\_head](#)\* [list\\_head::next](#)

3.27.1.2 struct [list\\_head](#) \* [list\\_head::prev](#)

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

- include/[elist.h](#)

## 3.28 ncld\_fh Struct Reference

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

### Data Fields

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

### 3.28.1 Field Documentation

3.28.1.1 int [ncld\\_fh::errc](#)

3.28.1.2 void\* [ncld\\_fh::event\\_arg](#)

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



3.28.1.4 unsigned int ncld\_fh::event\_mask

3.28.1.5 struct cldc\_fh\* ncld\_fh::fh

3.28.1.6 bool ncld\_fh::is\_open

3.28.1.7 int ncld\_fh::nios

3.28.1.8 struct ncld\_sess\* ncld\_fh::sess

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

- include/[ncld.h](#)

## 3.29 ncld\_read Struct Reference

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

### Data Fields

- const void \* [ptr](#)
- long [length](#)
- struct [cldc\\_node\\_metadata](#) meta
- struct [ncld\\_fh](#) \* fh
- bool [is\\_done](#)
- int [errc](#)

### 3.29.1 Field Documentation

3.29.1.1 int ncld\_read::errc

3.29.1.2 struct ncld\_fh\* ncld\_read::fh

3.29.1.3 bool ncld\_read::is\_done

3.29.1.4 long ncld\_read::length

3.29.1.5 struct cldc\_node\_metadata ncld\_read::meta

3.29.1.6 const void\* ncld\_read::ptr

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

- include/[ncld.h](#)

### 3.30 ncld\_sess Struct Reference

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

#### Data Fields

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

#### 3.30.1 Field Documentation

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

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

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

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

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

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

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

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

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

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

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

3.30.1.12 struct `cld_timer_list` `ncld_sess::tlist`

3.30.1.13 int `ncld_sess::to_thread[2]`

3.30.1.14 struct `cldc_udp*` `ncld_sess::udp`

3.30.1.15 struct `cld_timer` `ncld_sess::udp_timer`

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

- include/[ncld.h](#)

## 3.31 objcache Struct Reference

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

### Data Fields

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

### 3.31.1 Field Documentation

3.31.1.1 GMutex\* `objcache::lock`

3.31.1.2 GHashTable\* `objcache::table`

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

- include/[objcache.h](#)

## 3.32 objcache\_entry Struct Reference

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

### Data Fields

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

### 3.32.1 Field Documentation

3.32.1.1 unsigned int objcache\_entry::flags

3.32.1.2 unsigned int objcache\_entry::hash

3.32.1.3 int objcache\_entry::ref

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

- include/[objcache.h](#)

## 3.33 st\_client Struct Reference

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

### Data Fields

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

### 3.33.1 Field Documentation

3.33.1.1 int st\_client::fd

3.33.1.2 char\* st\_client::host

3.33.1.3 char\* st\_client::key

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

3.33.1.5 SSL\* st\_client::ssl

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

3.33.1.7 char\* st\_client::user

3.33.1.8 bool st\_client::verbose

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

- include/[chunkc.h](#)

## 3.34 `st_keylist` Struct Reference

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

### Data Fields

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

#### 3.34.1 Field Documentation

3.34.1.1 GList\* `st_keylist::contents`

3.34.1.2 char\* `st_keylist::name`

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

- include/[chunkc.h](#)

## 3.35 `st_object` Struct Reference

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

### Data Fields

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

#### 3.35.1 Field Documentation

3.35.1.1 char\* `st_object::etag`

3.35.1.2 char\* `st_object::name`

3.35.1.3 char\* `st_object::owner`

3.35.1.4 `uint64_t st_object::size`

3.35.1.5 `char* st_object::time_mod`

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

- `include/chunkc.h`

## Chapter 4

# File Documentation

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

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

#### Defines

- #define [MDB\\_TPATH\\_FMT](#) "%s/%X"
- #define [BAD\\_TPATH\\_FMT](#) "%s/bad"
- #define [PREFIX\\_LEN](#) 3

#### 4.1.1 Define Documentation

4.1.1.1 #define [BAD\\_TPATH\\_FMT](#) "%s/bad"

4.1.1.2 #define [MDB\\_TPATH\\_FMT](#) "%s/%X"

4.1.1.3 #define [PREFIX\\_LEN](#) 3

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

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

#### Data Structures

- struct [chunksrv\\_req](#)
- struct [chunksrv\\_resp](#)
- struct [chunksrv\\_resp\\_get](#)
- struct [chunk\\_check\\_status](#)
- struct [chunksrv\\_resp\\_chkstat](#)

## Defines

- #define `CHUNKD_MAGIC` "CHUNKDv1"

## Enumerations

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

### 4.2.1 Define Documentation

#### 4.2.1.1 #define `CHUNKD_MAGIC` "CHUNKDv1"

### 4.2.2 Enumeration Type Documentation

#### 4.2.2.1 anonymous enum

Enumerator:

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

#### 4.2.2.2 enum `chunk_check_state`

Enumerator:

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



## 4.2.2.3 enum chunk\_errcode

Enumerator:

*che\_Success*  
*che\_AccessDenied*  
*che\_InternalError*  
*che\_InvalidArgument*  
*che\_InvalidURI*  
*che\_NoSuchKey*  
*che\_SignatureDoesNotMatch*  
*che\_InvalidKey*  
*che\_InvalidTable*  
*che\_Busy*  
*che\_KeyExists*

## 4.2.2.4 enum chunk\_flags

Enumerator:

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

## 4.2.2.5 enum chunksrv\_ops

Enumerator:

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

### 4.3 include/chunkc.h File Reference

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

#### Data Structures

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

#### Functions

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

### 4.3.1 Function Documentation

- 4.3.1.1 `bool stc_check_start ( struct st_client * stc )`
- 4.3.1.2 `bool stc_check_status ( struct st_client * stc, struct chunk_check_status * out )`
- 4.3.1.3 `bool stc_cp ( struct st_client * stc, const void * dest_key, size_t dest_key_len, const void * src_key, size_t src_key_len )`
- 4.3.1.4 `bool stc_del ( struct st_client * stc, const void * key, size_t key_len )`
- 4.3.1.5 `void stc_free ( struct st_client * stc )`
- 4.3.1.6 `void stc_free_keylist ( struct st_keylist * keylist )`
- 4.3.1.7 `void stc_free_object ( struct st_object * obj )`
- 4.3.1.8 `bool stc_get ( struct st_client * stc, const void * key, size_t key_len, size_t(*)(void *, size_t, size_t, void *) write_cb, void * user_data )`
- 4.3.1.9 `void* stc_get_inline ( struct st_client * stc, const void * key, size_t key_len, size_t * len )`
- 4.3.1.10 `size_t stc_get_recv ( struct st_client * stc, void * data, size_t len )`
- 4.3.1.11 `bool stc_get_start ( struct st_client * stc, const void * key, size_t key_len, int * pfd, uint64_t * len )`
- 4.3.1.12 `void stc_init ( void )`
- 4.3.1.13 `struct st_keylist* stc_keys ( struct st_client * stc )` [read]
- 4.3.1.14 `struct st_client* stc_new ( const char * service_host, int port, const char * user, const char * secret_key, bool encrypt )` [read]
- 4.3.1.15 `bool stc_ping ( struct st_client * stc )`
- 4.3.1.16 `bool stc_put ( struct st_client * stc, const void * key, size_t key_len, size_t(*)(void *, size_t, size_t, void *) read_cb, uint64_t len, void * user_data, uint32_t flags )`
- 4.3.1.17 `bool stc_put_inline ( struct st_client * stc, const void * key, size_t key_len, void * data, uint64_t len, uint32_t flags )`
- 4.3.1.18 `size_t stc_put_send ( struct st_client * stc, void * data, size_t len )`
- 4.3.1.19 `bool stc_put_start ( struct st_client * stc, const void * key, size_t key_len, uint64_t cont_len, int * pfd, uint32_t flags )`

4.3.1.20 `bool stc_put_sync ( struct st_client * stc )`

4.3.1.21 `int stc_readport ( const char * fname )`

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

## 4.4 include/chunksrv.h File Reference

```
#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 "%016llx"`
- `#define SIDARG(sid) cld_sid2llu(sid)`

- #define `CLD_PKT_FTR_LEN` `sizeof(struct cld_pkt_ftr)`  
*Length of the packet footer.*
- #define `PKT_HDR_TO_STR_SCRATCH_LEN` 128

## Functions

- void `cld_timer_add` (struct `cld_timer_list` \*tlist, struct `cld_timer` \*timer, time\_t expires)
- void `cld_timer_del` (struct `cld_timer_list` \*tlist, struct `cld_timer` \*timer)
- time\_t `cld_timers_run` (struct `cld_timer_list` \*tlist)
- unsigned long long `cld_sid2llu` (const uint8\_t \*sid)
- void `cld_rand64` (void \*p)
- const char \* `cld_errstr` (enum `cld_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

<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](#)

### Defines

- #define [LIST\\_HEAD\\_INIT](#)(name) { &(name), &(name) }
- #define [LIST\\_HEAD](#)(name) struct [list\\_head](#) name = [LIST\\_HEAD\\_INIT](#)(name)
- #define [INIT\\_LIST\\_HEAD](#)(ptr)
- #define [list\\_entry](#)(ptr, type, member) ((type \*)((char \*) (ptr) - (unsigned long)(&((type \*)0)->member)))  
*list\_entry* - get the struct for this entry : the &struct [list\\_head](#) pointer.
- #define [list\\_for\\_each](#)(pos, head)  
*list\_for\_each* - iterate over a list : the &struct [list\\_head](#) to use as a loop counter.
- #define [list\\_for\\_each\\_prev](#)(pos, head)  
*list\_for\_each\_prev* - iterate over a list backwards : the &struct [list\\_head](#) to use as a loop counter.
- #define [list\\_for\\_each\\_safe](#)(pos, n, head)  
*list\_for\_each\_safe* - iterate over a list safe against removal of list entry : the &struct [list\\_head](#) to use as a loop counter.
- #define [list\\_for\\_each\\_entry](#)(pos, head, member)  
*list\_for\_each\_entry* - iterate over list of given type : the type \* to use as a loop counter.
- #define [list\\_for\\_each\\_entry\\_safe](#)(pos, n, head, member)  
*list\_for\_each\_entry\_safe* - iterate over list of given type safe against removal of list entry : the type \* to use as a loop counter.
- #define [list\\_for\\_each\\_entry\\_continue](#)(pos, head, member)  
*list\_for\_each\_entry\_continue* - iterate over list of given type continuing after existing point : the type \* to use as a loop counter.

### 4.8.1 Define Documentation

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

##### Value:

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

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

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

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

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

**Value:**

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

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

: the head for your list.

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

**Value:**

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

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

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

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

**Value:**

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

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

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

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

**Value:**

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

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

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

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

**Value:**

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

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

: the head for your list.

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

**Value:**

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

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

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

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

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

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

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

### Data Structures

- struct `hail_log`

## Defines

- #define [ATTR\\_PRINTF](#)(x, y)
- #define [HAIL\\_VERBOSE](#)(log,...)
 

*Print out a CLD session debug message if enabled.*
- #define [HAIL\\_DEBUG](#)(log,...)
 

*Print out an application debug message if enabled.*
- #define [HAIL\\_INFO](#)(log,...) (log)->func(LOG\_INFO, \_\_VA\_ARGS\_\_)
 

*Print out an informational log message.*
- #define [HAIL\\_WARN](#)(log,...) (log)->func(LOG\_WARNING, \_\_VA\_ARGS\_\_)
 

*Print out a warning message.*
- #define [HAIL\\_ERR](#)(log,...) (log)->func(LOG\_ERR, \_\_VA\_ARGS\_\_)
 

*Print out an error message.*
- #define [HAIL\\_CRIT](#)(log,...) (log)->func(LOG\_CRIT, \_\_VA\_ARGS\_\_)
 

*Print out a critical warning message.*

### 4.9.1 Define Documentation

4.9.1.1 #define [ATTR\\_PRINTF](#)( x, y )

4.9.1.2 #define [HAIL\\_CRIT](#)( log, ... ) (log)->func(LOG\_CRIT, \_\_VA\_ARGS\_\_)

Print out a critical warning message.

4.9.1.3 #define [HAIL\\_DEBUG](#)( log, ... )

**Value:**

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

Print out an application debug message if enabled.

4.9.1.4 #define [HAIL\\_ERR](#)( log, ... ) (log)->func(LOG\_ERR, \_\_VA\_ARGS\_\_)

Print out an error message.

4.9.1.5 #define [HAIL\\_INFO](#)( log, ... ) (log)->func(LOG\_INFO, \_\_VA\_ARGS\_\_)

Print out an informational log message.

#### 4.9.1.6 #define HAIL\_VERBOSE( log, ... )

##### Value:

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

Print out a CLD session debug message if enabled.

#### 4.9.1.7 #define HAIL\_WARN( log, ... ) (log)->func(LOG\_WARNING, \_\_VA\_ARGS\_\_)

Print out a warning message.

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

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

### Functions

- u\_long [xdr\\_sizeof](#) (xdrproc\_t, void \*)

#### 4.10.1 Function Documentation

##### 4.10.1.1 u\_long xdr\_sizeof ( xdrproc\_t , void \* )

## 4.11 include/hstor.h File Reference

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

### Data Structures

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

## Defines

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

## Enumerations

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

## Functions

- char \* [hutil\\_time2str](#) (char \*buf, int len, time\_t time)
- time\_t [hutil\\_str2time](#) (const char \*timestr)
- int [hreq\\_hdr\\_push](#) (struct [http\\_req](#) \*req, char \*key, char \*val)
- char \* [hreq\\_hdr](#) (struct [http\\_req](#) \*req, const char \*key)
- void [hreq\\_sign](#) (struct [http\\_req](#) \*req, const char \*bucket, const char \*key, char \*b64hmac\_out)
- GHashTable \* [hreq\\_query](#) (struct [http\\_req](#) \*req)
- int [hreq\\_is\\_query](#) (struct [http\\_req](#) \*req)
- void [hreq\\_free](#) (struct [http\\_req](#) \*req)
- int [hreq\\_acl\\_canned](#) (struct [http\\_req](#) \*req)
- struct [http\\_uri](#) \* [huri\\_parse](#) (struct [http\\_uri](#) \*uri\_dest, char \*uri\_src\_text)
- int [huri\\_field\\_unescape](#) (char \*s, int s\_len)
- char \* [huri\\_field\\_escape](#) (char \*signed\_str, unsigned char mask)
- void [hstor\\_free](#) (struct [hstor\\_client](#) \*hstor)
- void [hstor\\_free\\_blist](#) (struct [hstor\\_blist](#) \*blist)
- void [hstor\\_free\\_bucket](#) (struct [hstor\\_bucket](#) \*buck)
- void [hstor\\_free\\_object](#) (struct [hstor\\_object](#) \*obj)
- void [hstor\\_free\\_keylist](#) (struct [hstor\\_keylist](#) \*keylist)
- struct [hstor\\_client](#) \* [hstor\\_new](#) (const char \*service\_acc, const char \*service\_host, const char \*user, const char \*secret\_key)
- bool [hstor\\_add\\_bucket](#) (struct [hstor\\_client](#) \*hstor, const char \*name)
- bool [hstor\\_del\\_bucket](#) (struct [hstor\\_client](#) \*hstor, const char \*name)
- struct [hstor\\_blist](#) \* [hstor\\_list\\_buckets](#) (struct [hstor\\_client](#) \*hstor)
- bool [hstor\\_get](#) (struct [hstor\\_client](#) \*hstor, const char \*bucket, const char \*key, size\_t(\*write\_cb)(void \*, size\_t, size\_t, void \*), void \*user\_data, bool want\_headers)
- void \* [hstor\\_get\\_inline](#) (struct [hstor\\_client](#) \*hstor, const char \*bucket, const char \*key, bool want\_headers, size\_t \*len)
- bool [hstor\\_put](#) (struct [hstor\\_client](#) \*hstor, const char \*bucket, const char \*key, size\_t(\*read\_cb)(void \*, size\_t, size\_t, void \*), uint64\_t len, void \*user\_data, char \*\*user\_hdrs)

- bool [hstor\\_put\\_inline](#) (struct [hstor\\_client](#) \*hstor, const char \*bucket, const char \*key, void \*data, uint64\_t len, char \*\*user\_hdrs)
- bool [hstor\\_del](#) (struct [hstor\\_client](#) \*hstor, const char \*bucket, const char \*key)
- struct [hstor\\_keylist](#) \* [hstor\\_keys](#) (struct [hstor\\_client](#) \*hstor, const char \*bucket, const char \*prefix, const char \*marker, const char \*delim, unsigned int max\_keys)

#### 4.11.1 Define Documentation

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

4.11.1.2 `#define PATH_ESCAPE_MASK 0x02`

4.11.1.3 `#define QUERY_ESCAPE_MASK 0x04`

#### 4.11.2 Enumeration Type Documentation

4.11.2.1 anonymous enum

Enumerator:

***HREQ\_MAX\_HDR***

4.11.2.2 enum ReqACLC

Enumerator:

***ACLC\_PRIV***

***ACLC\_PUB\_R***

***ACLC\_PUB\_RW***

***ACLC\_AUTH\_R***

***ACLCNUM***

4.11.2.3 enum ReqQ

Enumerator:

***URIQ\_ACL***

***URIQ\_LOCATION***

***URIQ\_LOGGING***

***URIQ\_TORRENT***

***URIQNUM***



### 4.11.3 Function Documentation

- 4.11.3.1 int hreq\_acl\_canned ( struct http\_req \* req )
- 4.11.3.2 void hreq\_free ( struct http\_req \* req )
- 4.11.3.3 char\* hreq\_hdr ( struct http\_req \* req, const char \* key )
- 4.11.3.4 int hreq\_hdr\_push ( struct http\_req \* req, char \* key, char \* val )
- 4.11.3.5 int hreq\_is\_query ( struct http\_req \* req )
- 4.11.3.6 GHashTable\* hreq\_query ( struct http\_req \* req )
- 4.11.3.7 void hreq\_sign ( struct http\_req \* req, const char \* bucket, const char \* key, char \* b64hmac\_out )
- 4.11.3.8 bool hstor\_add\_bucket ( struct hstor\_client \* hstor, const char \* name )
- 4.11.3.9 bool hstor\_del ( struct hstor\_client \* hstor, const char \* bucket, const char \* key )
- 4.11.3.10 bool hstor\_del\_bucket ( struct hstor\_client \* hstor, const char \* name )
- 4.11.3.11 void hstor\_free ( struct hstor\_client \* hstor )
- 4.11.3.12 void hstor\_free\_blist ( struct hstor\_blist \* blist )
- 4.11.3.13 void hstor\_free\_bucket ( struct hstor\_bucket \* buck )
- 4.11.3.14 void hstor\_free\_keylist ( struct hstor\_keylist \* keylist )
- 4.11.3.15 void hstor\_free\_object ( struct hstor\_object \* obj )
- 4.11.3.16 bool hstor\_get ( struct hstor\_client \* hstor, const char \* bucket, const char \* key, size\_t\*(void \*, size\_t, size\_t, void \*) write\_cb, void \* user\_data, bool want\_headers )
- 4.11.3.17 void\* hstor\_get\_inline ( struct hstor\_client \* hstor, const char \* bucket, const char \* key, bool want\_headers, size\_t \* len )
- 4.11.3.18 struct hstor\_keylist\* hstor\_keys ( struct hstor\_client \* hstor, const char \* bucket, const char \* prefix, const char \* marker, const char \* delim, unsigned int max\_keys ) [read]
- 4.11.3.19 struct hstor\_blist\* hstor\_list\_buckets ( struct hstor\_client \* hstor ) [read]
- 4.11.3.20 struct hstor\_client\* hstor\_new ( const char \* service\_acc, const char \* service\_host, const char \* user, const char \* secret\_key ) [read]

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

## 4.12 include/ncl.d.h File Reference

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

### Data Structures

- struct [ncl.d\\_sess](#)
- struct [ncl.d\\_fh](#)
- struct [ncl.d\\_read](#)

### Functions

- struct [ncl.d\\_sess](#) \* [ncl.d\\_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 [ncl.d\\_fh](#) \* [ncl.d\\_open](#) (struct [ncl.d\\_sess](#) \*s, const char \*fname, unsigned int mode, int \*error, unsigned int events, void(\*event)(void \*, unsigned int), void \*ev\_arg)
- int [ncl.d\\_del](#) (struct [ncl.d\\_sess](#) \*nsess, const char \*fname)
- struct [ncl.d\\_read](#) \* [ncl.d\\_get](#) (struct [ncl.d\\_fh](#) \*fh, int \*error)
- struct [ncl.d\\_read](#) \* [ncl.d\\_get\\_meta](#) (struct [ncl.d\\_fh](#) \*fh, int \*error)
- void [ncl.d\\_read\\_free](#) (struct [ncl.d\\_read](#) \*rp)
- int [ncl.d\\_write](#) (struct [ncl.d\\_fh](#) \*, const void \*data, long len)
- int [ncl.d\\_trylock](#) (struct [ncl.d\\_fh](#) \*)
- int [ncl.d\\_qlock](#) (struct [ncl.d\\_fh](#) \*)
- int [ncl.d\\_unlock](#) (struct [ncl.d\\_fh](#) \*)
- void [ncl.d\\_close](#) (struct [ncl.d\\_fh](#) \*)

- void [ncld\\_sess\\_close](#) (struct [ncld\\_sess](#) \*s)
- void [ncld\\_init](#) (void)

### 4.12.1 Function Documentation

4.12.1.1 void [ncld\\_close](#) ( struct [ncld\\_fh](#) \* )

4.12.1.2 int [ncld\\_del](#) ( struct [ncld\\_sess](#) \* *nsess*, const char \* *fname* )

4.12.1.3 struct [ncld\\_read](#)\* [ncld\\_get](#) ( struct [ncld\\_fh](#) \* *fh*, int \* *error* ) [read]

4.12.1.4 struct [ncld\\_read](#)\* [ncld\\_get\\_meta](#) ( struct [ncld\\_fh](#) \* *fh*, int \* *error* ) [read]

4.12.1.5 void [ncld\\_init](#) ( void )

4.12.1.6 struct [ncld\\_fh](#)\* [ncld\\_open](#) ( struct [ncld\\_sess](#) \* *s*, const char \* *fname*, unsigned int *mode*, int \* *error*, unsigned int *events*, void(\*) (void \*, unsigned int) *event*, void \* *ev\_arg* ) [read]

4.12.1.7 int [ncld\\_qlock](#) ( struct [ncld\\_fh](#) \* )

4.12.1.8 void [ncld\\_read\\_free](#) ( struct [ncld\\_read](#) \* *rp* )

4.12.1.9 void [ncld\\_sess\\_close](#) ( struct [ncld\\_sess](#) \* *s* )

4.12.1.10 struct [ncld\\_sess](#)\* [ncld\\_sess\\_open](#) ( const char \* *host*, int *port*, int \* *error*, void(\*) (void \*, unsigned int) *event*, void \* *ev\_arg*, const char \* *cld\_user*, const char \* *cld\_key*, struct [hail\\_log](#) \* *log* ) [read]

4.12.1.11 int [ncld\\_trylock](#) ( struct [ncld\\_fh](#) \* )

4.12.1.12 int [ncld\\_unlock](#) ( struct [ncld\\_fh](#) \* )

4.12.1.13 int [ncld\\_write](#) ( struct [ncld\\_fh](#) \* , const void \* *data*, long *len* )

## 4.13 include/objcache.h File Reference

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

### Data Structures

- struct [objcache](#)
- struct [objcache\\_entry](#)

## 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* )