```
**********************************************************
    4202 Tue Jan 15 15:38:21 2013
new/usr/src/common/zfs/zfeature_common.c
%B
**********************************************************
```

```
 1 /*
 2  * CDDL HEADER START
 3  *
 4  * The contents of this file are subject to the terms of the
 5  * Common Development and Distribution License (the "License").
 6  * You may not use this file except in compliance with the License.
 7  *
 8  * You can obtain a copy of the license at usr/src/OPENSOLARIS.LICENSE
 9  * or http://www.opensolaris.org/os/licensing.
10  * See the License for the specific language governing permissions
11  * and limitations under the License.
12  *
13  * When distributing Covered Code, include this CDDL HEADER in each
14  * file and include the License file at usr/src/OPENSOLARIS.LICENSE.
15  * If applicable, add the following below this CDDL HEADER, with the
16  * fields enclosed by brackets "[]" replaced with your own identifying
17  * information: Portions Copyright [yyyy] [name of copyright owner]
18  *
19  * CDDL HEADER END
20  */

22 /*
23  * Copyright (c) 2012 by Delphix. All rights reserved.
24  * Copyright (c) 2013 by Saso Kiselkov. All rights reserved.
25  */

27 #ifdef _KERNEL
28 #include <sys/systm.h>
29 #else
30 #include <errno.h>
31 #include <string.h>
32 #endif
33 #include <sys/debug.h>
34 #include <sys/fs/zfs.h>
35 #include <sys/inttypes.h>
36 #include <sys/types.h>
37 #include "zfeature_common.h"

39 /*
40  * Set to disable all feature checks while opening pools, allowing pools with
41  * unsupported features to be opened. Set for testing only.
42  */
43 boolean_t zfeature_checks_disable = B_FALSE;

45 zfeature_info_t spa_feature_table[SPA_FEATURES];

47 /*
48  * Valid characters for feature guids. This list is mainly for aesthetic
49  * purposes and could be expanded in the future. There are different allowed
50  * characters in the guids reverse dns portion (before the colon) and its
51  * short name (after the colon).
52  */
53 static int
54 valid_char(char c, boolean_t after_colon)
55 {
56         return ((c >= 'a' && c <= 'z') ||
57             (c >= '0' && c <= '9') ||
58             c == (after_colon ? '_' : '.'));
59 }
```

_____***unchanged_portion_omitted_***


```
151 void
152 zpool_feature_init(void)
153 {
154         zfeature_register(SPA_FEATURE_ASYNC_DESTROY,
155             "com.delphix:async_destroy", "async_destroy",
156             "Destroy filesystems asynchronously.", B_TRUE, B_FALSE, NULL);
157         zfeature_register(SPA_FEATURE_EMPTY_BPOBJ,
158             "com.delphix:empty_bpobj", "empty_bpobj",
159             "Snapshots use less space.", B_TRUE, B_FALSE, NULL);
160         zfeature_register(SPA_FEATURE_LZ4_COMPRESS,
161             "org.illumos:lz4_compress", "lz4_compress",
162             "LZ4 compression algorithm support.", B_FALSE, B_FALSE, NULL);
163 }
```

_____***unchanged_portion_omitted_***

```
**********************************************************
    2178 Tue Jan 15 15:38:21 2013
new/usr/src/common/zfs/zfeature_common.h
%B
**********************************************************
```

```
/*
 * CDDL HEADER START
 *
 * The contents of this file are subject to the terms of the
 * Common Development and Distribution License (the "License").
 * You may not use this file except in compliance with the License.
 *
 * You can obtain a copy of the license at usr/src/OPENSOLARIS.LICENSE
 * or http://www.opensolaris.org/os/licensing.
 * See the License for the specific language governing permissions
 * and limitations under the License.
 *
 * When distributing Covered Code, include this CDDL HEADER in each
 * file and include the License file at usr/src/OPENSOLARIS.LICENSE.
 * If applicable, add the following below this CDDL HEADER, with the
 * fields enclosed by brackets "[]" replaced with your own identifying
 * information: Portions Copyright [yyyy] [name of copyright owner]
 *
 * CDDL HEADER END
 */

/*
 * Copyright (c) 2012 by Delphix. All rights reserved.
 * Copyright (c) 2013 by Saso Kiselkov. All rights reserved.
 */

#ifndef _ZFEATURE_COMMON_H
#define _ZFEATURE_COMMON_H

#include <sys/fs/zfs.h>
#include <sys/inttypes.h>
#include <sys/types.h>

#ifdef  __cplusplus
extern "C" {
#endif

struct zfeature_info;

typedef struct zfeature_info {
        const char *fi_uname;   /* User-facing feature name */
        const char *fi_guid;    /* On-disk feature identifier */
        const char *fi_desc;    /* Feature description */
        boolean_t fi_can_readonly; /* Can open pool readonly w/o support? */
        boolean_t fi_mos;       /* Is the feature necessary to read the MOS? */
        struct zfeature_info **fi_depends; /* array; null terminated */
} zfeature_info_t;

typedef int (zfeature_func_t)(zfeature_info_t *fi, void *arg);

#define ZFS_FEATURE_DEBUG

enum spa_feature {
        SPA_FEATURE_ASYNC_DESTROY,
        SPA_FEATURE_EMPTY_BPOBJ,
        SPA_FEATURE_LZ4_COMPRESS,
        SPA_FEATURES
} spa_feature_t;
```

*_____unchanged_portion_omitted_*

```
**********************************************************
   19834 Tue Jan 15 15:38:21 2013
new/usr/src/common/zfs/zfs_prop.c
%B
**********************************************************
```

```
/*
 * CDDL HEADER START
 *
 * The contents of this file are subject to the terms of the
 * Common Development and Distribution License (the "License").
 * You may not use this file except in compliance with the License.
 *
 * You can obtain a copy of the license at usr/src/OPENSOLARIS.LICENSE
 * or http://www.opensolaris.org/os/licensing.
 * See the License for the specific language governing permissions
 * and limitations under the License.
 *
 * When distributing Covered Code, include this CDDL HEADER in each
 * file and include the License file at usr/src/OPENSOLARIS.LICENSE.
 * If applicable, add the following below this CDDL HEADER, with the
 * fields enclosed by brackets "[]" replaced with your own identifying
 * information: Portions Copyright [yyyy] [name of copyright owner]
 *
 * CDDL HEADER END
 */
/*
 * Copyright (c) 2005, 2010, Oracle and/or its affiliates. All rights reserved.
 * Copyright (c) 2011 by Delphix. All rights reserved.
 * Copyright (c) 2013 by Saso Kiselkov. All rights reserved.
 */

/* Portions Copyright 2010 Robert Milkowski */

#include <sys/zio.h>
#include <sys/spa.h>
#include <sys/u8_textprep.h>
#include <sys/zfs_acl.h>
#include <sys/zfs_ioctl.h>
#include <sys/zfs_znode.h>

#include "zfs_prop.h"
#include "zfs_deleg.h"

#if defined(_KERNEL)
#include <sys/systm.h>
#else
#include <stdlib.h>
#include <string.h>
#include <ctype.h>
#endif

static zprop_desc_t zfs_prop_table[ZFS_NUM_PROPS];

/* Note this is indexed by zfs_userquota_prop_t, keep the order the same */
const char *zfs_userquota_prop_prefixes[] = {
        "userused@",
        "userquota@",
        "groupused@",
        "groupquota@"
};
```

*unchanged_portion_omitted_*

```
void
zfs_prop_init(void)
{
        static zprop_index_t checksum_table[] = {
```



```
                { "on",         ZIO_CHECKSUM_ON },
                { "off",        ZIO_CHECKSUM_OFF },
                { "fletcher2",  ZIO_CHECKSUM_FLETCHER_2 },
                { "fletcher4",  ZIO_CHECKSUM_FLETCHER_4 },
                { "sha256",     ZIO_CHECKSUM_SHA256 },
                { NULL }
        };

        static zprop_index_t dedup_table[] = {
                { "on",         ZIO_CHECKSUM_ON },
                { "off",        ZIO_CHECKSUM_OFF },
                { "verify",     ZIO_CHECKSUM_ON | ZIO_CHECKSUM_VERIFY },
                { "sha256",     ZIO_CHECKSUM_SHA256 },
                { "sha256,verify",
                                ZIO_CHECKSUM_SHA256 | ZIO_CHECKSUM_VERIFY },
                { NULL }
        };

        static zprop_index_t compress_table[] = {
                { "on",         ZIO_COMPRESS_ON },
                { "off",        ZIO_COMPRESS_OFF },
                { "lzjb",       ZIO_COMPRESS_LZJB },
                { "gzip",       ZIO_COMPRESS_GZIP_6 },  /* gzip default */
                { "gzip-1",     ZIO_COMPRESS_GZIP_1 },
                { "gzip-2",     ZIO_COMPRESS_GZIP_2 },
                { "gzip-3",     ZIO_COMPRESS_GZIP_3 },
                { "gzip-4",     ZIO_COMPRESS_GZIP_4 },
                { "gzip-5",     ZIO_COMPRESS_GZIP_5 },
                { "gzip-6",     ZIO_COMPRESS_GZIP_6 },
                { "gzip-7",     ZIO_COMPRESS_GZIP_7 },
                { "gzip-8",     ZIO_COMPRESS_GZIP_8 },
                { "gzip-9",     ZIO_COMPRESS_GZIP_9 },
                { "zle",        ZIO_COMPRESS_ZLE },
                { "lz4",        ZIO_COMPRESS_LZ4 },
                { "lz4hc",      ZIO_COMPRESS_LZ4HC },
                { NULL }
        };

        static zprop_index_t snapdir_table[] = {
                { "hidden",     ZFS_SNAPDIR_HIDDEN },
                { "visible",    ZFS_SNAPDIR_VISIBLE },
                { NULL }
        };

        static zprop_index_t acl_mode_table[] = {
                { "discard",    ZFS_ACL_DISCARD },
                { "groupmask",  ZFS_ACL_GROUPMASK },
                { "passthrough", ZFS_ACL_PASSTHROUGH },
                { "restricted", ZFS_ACL_RESTRICTED },
                { NULL }
        };

        static zprop_index_t acl_inherit_table[] = {
                { "discard",    ZFS_ACL_DISCARD },
                { "noallow",    ZFS_ACL_NOALLOW },
                { "restricted", ZFS_ACL_RESTRICTED },
                { "passthrough", ZFS_ACL_PASSTHROUGH },
                { "secure",     ZFS_ACL_RESTRICTED }, /* bkwrd compatability */
                { "passthrough-x", ZFS_ACL_PASSTHROUGH_X },
                { NULL }
        };

        static zprop_index_t case_table[] = {
                { "sensitive",          ZFS_CASE_SENSITIVE },
                { "insensitive",        ZFS_CASE_INSENSITIVE },
                { "mixed",              ZFS_CASE_MIXED },
```

```
133                 { NULL }
134         };
135
136         static zprop_index_t copies_table[] = {
137                 { "1",          1 },
138                 { "2",          2 },
139                 { "3",          3 },
140                 { NULL }
141         };
142
143         /*
144          * Use the unique flags we have to send to u8_strcmp() and/or
145          * u8_textprep() to represent the various normalization property
146          * values.
147          */
148         static zprop_index_t normalize_table[] = {
149                 { "none",       0 },
150                 { "formD",      U8_TEXTPREP_NFD },
151                 { "formKC",     U8_TEXTPREP_NFKC },
152                 { "formC",      U8_TEXTPREP_NFC },
153                 { "formKD",     U8_TEXTPREP_NFKD },
154                 { NULL }
155         };
156
157         static zprop_index_t version_table[] = {
158                 { "1",          1 },
159                 { "2",          2 },
160                 { "3",          3 },
161                 { "4",          4 },
162                 { "5",          5 },
163                 { "current",    ZPL_VERSION },
164                 { NULL }
165         };
166
167         static zprop_index_t boolean_table[] = {
168                 { "off",        0 },
169                 { "on",         1 },
170                 { NULL }
171         };
172
173         static zprop_index_t logbias_table[] = {
174                 { "latency",    ZFS_LOGBIAS_LATENCY },
175                 { "throughput", ZFS_LOGBIAS_THROUGHPUT },
176                 { NULL }
177         };
178
179         static zprop_index_t canmount_table[] = {
180                 { "off",        ZFS_CANMOUNT_OFF },
181                 { "on",         ZFS_CANMOUNT_ON },
182                 { "noauto",     ZFS_CANMOUNT_NOAUTO },
183                 { NULL }
184         };
185
186         static zprop_index_t cache_table[] = {
187                 { "none",       ZFS_CACHE_NONE },
188                 { "metadata",   ZFS_CACHE_METADATA },
189                 { "all",        ZFS_CACHE_ALL },
190                 { NULL }
191         };
192
193         static zprop_index_t sync_table[] = {
194                 { "standard",   ZFS_SYNC_STANDARD },
195                 { "always",     ZFS_SYNC_ALWAYS },
196                 { "disabled",   ZFS_SYNC_DISABLED },
197                 { NULL }
198         };
```



```
200         /* inherit index properties */
201         zprop_register_index(ZFS_PROP_SYNC, "sync", ZFS_SYNC_STANDARD,
202             PROP_INHERIT, ZFS_TYPE_FILESYSTEM | ZFS_TYPE_VOLUME,
203             "standard | always | disabled", "SYNC",
204             sync_table);
205         zprop_register_index(ZFS_PROP_CHECKSUM, "checksum",
206             ZIO_CHECKSUM_DEFAULT, PROP_INHERIT, ZFS_TYPE_FILESYSTEM |
207             ZFS_TYPE_VOLUME,
208             "on | off | fletcher2 | fletcher4 | sha256", "CHECKSUM",
209             checksum_table);
210         zprop_register_index(ZFS_PROP_DEDUP, "dedup", ZIO_CHECKSUM_OFF,
211             PROP_INHERIT, ZFS_TYPE_FILESYSTEM | ZFS_TYPE_VOLUME,
212             "on | off | verify | sha256[,verify]", "DEDUP",
213             dedup_table);
214         zprop_register_index(ZFS_PROP_COMPRESSION, "compression",
215             ZIO_COMPRESS_DEFAULT, PROP_INHERIT,
216             ZFS_TYPE_FILESYSTEM | ZFS_TYPE_VOLUME,
217             "on | off | lzjb | gzip | gzip-[1-9] | zle | lz4 | lz4hc",
218             "COMPRESS", compress_table);
214             "on | off | lzjb | gzip | gzip-[1-9] | zle", "COMPRESS",
215             compress_table);
219         zprop_register_index(ZFS_PROP_SNAPDIR, "snapdir", ZFS_SNAPDIR_HIDDEN,
220             PROP_INHERIT, ZFS_TYPE_FILESYSTEM,
221             "hidden | visible", "SNAPDIR", snapdir_table);
222         zprop_register_index(ZFS_PROP_ACLMODE, "aclmode", ZFS_ACL_DISCARD,
223             PROP_INHERIT, ZFS_TYPE_FILESYSTEM,
224             "discard | groupmask | passthrough | restricted", "ACLMODE",
225             acl_mode_table);
226         zprop_register_index(ZFS_PROP_ACLINHERIT, "aclinherit",
227             ZFS_ACL_RESTRICTED, PROP_INHERIT, ZFS_TYPE_FILESYSTEM,
228             "discard | noallow | restricted | passthrough | passthrough-x",
229             "ACLINHERIT", acl_inherit_table);
230         zprop_register_index(ZFS_PROP_COPIES, "copies", 1, PROP_INHERIT,
231             ZFS_TYPE_FILESYSTEM | ZFS_TYPE_VOLUME,
232             "1 | 2 | 3", "COPIES", copies_table);
233         zprop_register_index(ZFS_PROP_PRIMARYCACHE, "primarycache",
234             ZFS_CACHE_ALL, PROP_INHERIT,
235             ZFS_TYPE_FILESYSTEM | ZFS_TYPE_SNAPSHOT | ZFS_TYPE_VOLUME,
236             "all | none | metadata", "PRIMARYCACHE", cache_table);
237         zprop_register_index(ZFS_PROP_SECONDARYCACHE, "secondarycache",
238             ZFS_CACHE_ALL, PROP_INHERIT,
239             ZFS_TYPE_FILESYSTEM | ZFS_TYPE_SNAPSHOT | ZFS_TYPE_VOLUME,
240             "all | none | metadata", "SECONDARYCACHE", cache_table);
241         zprop_register_index(ZFS_PROP_LOGBIAS, "logbias", ZFS_LOGBIAS_LATENCY,
242             PROP_INHERIT, ZFS_TYPE_FILESYSTEM | ZFS_TYPE_VOLUME,
243             "latency | throughput", "LOGBIAS", logbias_table);
244
245         /* inherit index (boolean) properties */
246         zprop_register_index(ZFS_PROP_ATIME, "atime", 1, PROP_INHERIT,
247             ZFS_TYPE_FILESYSTEM, "on | off", "ATIME", boolean_table);
248         zprop_register_index(ZFS_PROP_DEVICES, "devices", 1, PROP_INHERIT,
249             ZFS_TYPE_FILESYSTEM | ZFS_TYPE_SNAPSHOT, "on | off", "DEVICES",
250             boolean_table);
251         zprop_register_index(ZFS_PROP_EXEC, "exec", 1, PROP_INHERIT,
252             ZFS_TYPE_FILESYSTEM | ZFS_TYPE_SNAPSHOT, "on | off", "EXEC",
253             boolean_table);
254         zprop_register_index(ZFS_PROP_SETUID, "setuid", 1, PROP_INHERIT,
255             ZFS_TYPE_FILESYSTEM | ZFS_TYPE_SNAPSHOT, "on | off", "SETUID",
256             boolean_table);
257         zprop_register_index(ZFS_PROP_READONLY, "readonly", 0, PROP_INHERIT,
258             ZFS_TYPE_FILESYSTEM | ZFS_TYPE_VOLUME, "on | off", "RDONLY",
259             boolean_table);
260         zprop_register_index(ZFS_PROP_ZONED, "zoned", 0, PROP_INHERIT,
261             ZFS_TYPE_FILESYSTEM, "on | off", "ZONED", boolean_table);
262         zprop_register_index(ZFS_PROP_XATTR, "xattr", 1, PROP_INHERIT,
```

```
263             ZFS_TYPE_FILESYSTEM | ZFS_TYPE_SNAPSHOT, "on | off", "XATTR",
264             boolean_table);
265         zprop_register_index(ZFS_PROP_VSCAN, "vscan", 0, PROP_INHERIT,
266             ZFS_TYPE_FILESYSTEM, "on | off", "VSCAN",
267             boolean_table);
268         zprop_register_index(ZFS_PROP_NBMAND, "nbmand", 0, PROP_INHERIT,
269             ZFS_TYPE_FILESYSTEM | ZFS_TYPE_SNAPSHOT, "on | off", "NBMAND",
270             boolean_table);

272         /* default index properties */
273         zprop_register_index(ZFS_PROP_VERSION, "version", 0, PROP_DEFAULT,
274             ZFS_TYPE_FILESYSTEM | ZFS_TYPE_SNAPSHOT,
275             "1 | 2 | 3 | 4 | 5 | current", "VERSION", version_table);
276         zprop_register_index(ZFS_PROP_CANMOUNT, "canmount", ZFS_CANMOUNT_ON,
277             PROP_DEFAULT, ZFS_TYPE_FILESYSTEM, "on | off | noauto",
278             "CANMOUNT", canmount_table);

280         /* readonly index (boolean) properties */
281         zprop_register_index(ZFS_PROP_MOUNTED, "mounted", 0, PROP_READONLY,
282             ZFS_TYPE_FILESYSTEM, "yes | no", "MOUNTED", boolean_table);
283         zprop_register_index(ZFS_PROP_DEFER_DESTROY, "defer_destroy", 0,
284             PROP_READONLY, ZFS_TYPE_SNAPSHOT, "yes | no", "DEFER_DESTROY",
285             boolean_table);

287         /* set once index properties */
288         zprop_register_index(ZFS_PROP_NORMALIZE, "normalization", 0,
289             PROP_ONETIME, ZFS_TYPE_FILESYSTEM | ZFS_TYPE_SNAPSHOT,
290             "none | formC | formD | formKC | formKD", "NORMALIZATION",
291             normalize_table);
292         zprop_register_index(ZFS_PROP_CASE, "casesensitivity",
293             ZFS_CASE_SENSITIVE, PROP_ONETIME, ZFS_TYPE_FILESYSTEM |
294             ZFS_TYPE_SNAPSHOT,
295             "sensitive | insensitive | mixed", "CASE", case_table);

297         /* set once index (boolean) properties */
298         zprop_register_index(ZFS_PROP_UTF8ONLY, "utf8only", 0, PROP_ONETIME,
299             ZFS_TYPE_FILESYSTEM | ZFS_TYPE_SNAPSHOT,
300             "on | off", "UTF8ONLY", boolean_table);

302         /* string properties */
303         zprop_register_string(ZFS_PROP_ORIGIN, "origin", NULL, PROP_READONLY,
304             ZFS_TYPE_FILESYSTEM | ZFS_TYPE_VOLUME, "<snapshot>", "ORIGIN");
305         zprop_register_string(ZFS_PROP_CLONES, "clones", NULL, PROP_READONLY,
306             ZFS_TYPE_SNAPSHOT, "<dataset>[,...]", "CLONES");
307         zprop_register_string(ZFS_PROP_MOUNTPOINT, "mountpoint", "/",
308             PROP_INHERIT, ZFS_TYPE_FILESYSTEM, "<path> | legacy | none",
309             "MOUNTPOINT");
310         zprop_register_string(ZFS_PROP_SHARENFS, "sharenfs", "off",
311             PROP_INHERIT, ZFS_TYPE_FILESYSTEM, "on | off | share(1M) options",
312             "SHARENFS");
313         zprop_register_string(ZFS_PROP_TYPE, "type", NULL, PROP_READONLY,
314             ZFS_TYPE_DATASET, "filesystem | volume | snapshot", "TYPE");
315         zprop_register_string(ZFS_PROP_SHARESMB, "sharesmb", "off",
316             PROP_INHERIT, ZFS_TYPE_FILESYSTEM,
317             "on | off | sharemgr(1M) options", "SHARESMB");
318         zprop_register_string(ZFS_PROP_MLSLABEL, "mlslabel",
319             ZFS_MLSLABEL_DEFAULT, PROP_INHERIT, ZFS_TYPE_DATASET,
320             "<sensitivity label>", "MLSLABEL");

322         /* readonly number properties */
323         zprop_register_number(ZFS_PROP_USED, "used", 0, PROP_READONLY,
324             ZFS_TYPE_DATASET, "<size>", "USED");
325         zprop_register_number(ZFS_PROP_AVAILABLE, "available", 0, PROP_READONLY,
326             ZFS_TYPE_FILESYSTEM | ZFS_TYPE_VOLUME, "<size>", "AVAIL");
327         zprop_register_number(ZFS_PROP_REFERENCED, "referenced", 0,
328             PROP_READONLY, ZFS_TYPE_DATASET, "<size>", "REFER");
```



```
329         zprop_register_number(ZFS_PROP_COMPRESSRATIO, "compressratio", 0,
330             PROP_READONLY, ZFS_TYPE_DATASET,
331             "<1.00x or higher if compressed>", "RATIO");
332         zprop_register_number(ZFS_PROP_REFRATIO, "refcompressratio", 0,
333             PROP_READONLY, ZFS_TYPE_DATASET,
334             "<1.00x or higher if compressed>", "REFRATIO");
335         zprop_register_number(ZFS_PROP_VOLBLOCKSIZE, "volblocksize",
336             ZVOL_DEFAULT_BLOCKSIZE, PROP_ONETIME,
337             ZFS_TYPE_VOLUME, "512 to 128k, power of 2", "VOLBLOCK");
338         zprop_register_number(ZFS_PROP_USEDSNAP, "usedbysnapshots", 0,
339             PROP_READONLY, ZFS_TYPE_FILESYSTEM | ZFS_TYPE_VOLUME, "<size>",
340             "USEDSNAP");
341         zprop_register_number(ZFS_PROP_USEDDS, "usedbydataset", 0,
342             PROP_READONLY, ZFS_TYPE_FILESYSTEM | ZFS_TYPE_VOLUME, "<size>",
343             "USEDDS");
344         zprop_register_number(ZFS_PROP_USEDCHILD, "usedbychildren", 0,
345             PROP_READONLY, ZFS_TYPE_FILESYSTEM | ZFS_TYPE_VOLUME, "<size>",
346             "USEDCHILD");
347         zprop_register_number(ZFS_PROP_USEDREFRESERV, "usedbyrefreservation", 0,
348             PROP_READONLY,
349             ZFS_TYPE_FILESYSTEM | ZFS_TYPE_VOLUME, "<size>", "USEDREFRESERV");
350         zprop_register_number(ZFS_PROP_USERREFS, "userrefs", 0, PROP_READONLY,
351             ZFS_TYPE_SNAPSHOT, "<count>", "USERREFS");
352         zprop_register_number(ZFS_PROP_WRITTEN, "written", 0, PROP_READONLY,
353             ZFS_TYPE_DATASET, "<size>", "WRITTEN");

355         /* default number properties */
356         zprop_register_number(ZFS_PROP_QUOTA, "quota", 0, PROP_DEFAULT,
357             ZFS_TYPE_FILESYSTEM, "<size> | none", "QUOTA");
358         zprop_register_number(ZFS_PROP_RESERVATION, "reservation", 0,
359             PROP_DEFAULT, ZFS_TYPE_FILESYSTEM | ZFS_TYPE_VOLUME,
360             "<size> | none", "RESERV");
361         zprop_register_number(ZFS_PROP_VOLSIZE, "volsize", 0, PROP_DEFAULT,
362             ZFS_TYPE_VOLUME, "<size>", "VOLSIZE");
363         zprop_register_number(ZFS_PROP_REFQUOTA, "refquota", 0, PROP_DEFAULT,
364             ZFS_TYPE_FILESYSTEM, "<size> | none", "REFQUOTA");
365         zprop_register_number(ZFS_PROP_REFRESERVATION, "refreservation", 0,
366             PROP_DEFAULT, ZFS_TYPE_FILESYSTEM | ZFS_TYPE_VOLUME,
367             "<size> | none", "REFRESERV");

369         /* inherit number properties */
370         zprop_register_number(ZFS_PROP_RECORDSIZE, "recordsize",
371             SPA_MAXBLOCKSIZE, PROP_INHERIT,
372             ZFS_TYPE_FILESYSTEM, "512 to 128k, power of 2", "RECSIZE");

374         /* hidden properties */
375         zprop_register_hidden(ZFS_PROP_CREATETXG, "createtxg", PROP_TYPE_NUMBER,
376             PROP_READONLY, ZFS_TYPE_DATASET, "CREATETXG");
377         zprop_register_hidden(ZFS_PROP_NUMCLONES, "numclones", PROP_TYPE_NUMBER,
378             PROP_READONLY, ZFS_TYPE_SNAPSHOT, "NUMCLONES");
379         zprop_register_hidden(ZFS_PROP_NAME, "name", PROP_TYPE_STRING,
380             PROP_READONLY, ZFS_TYPE_DATASET, "NAME");
381         zprop_register_hidden(ZFS_PROP_ISCSIOPTIONS, "iscsioptions",
382             PROP_TYPE_STRING, PROP_INHERIT, ZFS_TYPE_VOLUME, "ISCSIOPTIONS");
383         zprop_register_hidden(ZFS_PROP_STMF_SHAREINFO, "stmf_sbd_lu",
384             PROP_TYPE_STRING, PROP_INHERIT, ZFS_TYPE_VOLUME,
385             "STMF_SBD_LU");
386         zprop_register_hidden(ZFS_PROP_GUID, "guid", PROP_TYPE_NUMBER,
387             PROP_READONLY, ZFS_TYPE_DATASET, "GUID");
388         zprop_register_hidden(ZFS_PROP_USERACCOUNTING, "useraccounting",
389             PROP_TYPE_NUMBER, PROP_READONLY, ZFS_TYPE_DATASET,
390             "USERACCOUNTING");
391         zprop_register_hidden(ZFS_PROP_UNIQUE, "unique", PROP_TYPE_NUMBER,
392             PROP_READONLY, ZFS_TYPE_DATASET, "UNIQUE");
393         zprop_register_hidden(ZFS_PROP_OBJSETID, "objsetid", PROP_TYPE_NUMBER,
394             PROP_READONLY, ZFS_TYPE_DATASET, "OBJSETID");
```

```
396         /* oddball properties */
397         zprop_register_impl(ZFS_PROP_CREATION, "creation", PROP_TYPE_NUMBER, 0,
398             NULL, PROP_READONLY, ZFS_TYPE_DATASET,
399             "<date>", "CREATION", B_FALSE, B_TRUE, NULL);
400 }
```

unchanged_portion_omitted_

```
**********************************************************
   10798 Tue Jan 15 15:38:21 2013
new/usr/src/grub/grub-0.97/stage2/Makefile.am
%B
**********************************************************
```

```
  1 # For test target.
  2 TESTS = size_test
  3 noinst_SCRIPTS = $(TESTS)

  5 # For dist target.
  6 noinst_HEADERS = apic.h defs.h dir.h disk_inode.h disk_inode_ffs.h \
  7         fat.h filesys.h freebsd.h fs.h hercules.h i386-elf.h \
  8         imgact_aout.h iso9660.h jfs.h mb_header.h mb_info.h md5.h \
  9         nbi.h pc_slice.h serial.h shared.h smp-imps.h term.h \
 10         terminfo.h tparm.h nbi.h ufs.h ufs2.h vstafs.h xfs.h graphics.h \
 11         fsys_zfs.h
 12 EXTRA_DIST = setjmp.S apm.S $(noinst_SCRIPTS)

 14 # For <stage1.h>.
 15 INCLUDES = -I$(top_srcdir)/stage1

 17 # The library for /sbin/grub.
 18 noinst_LIBRARIES = libgrub.a
 19 libgrub_a_SOURCES = boot.c builtins.c char_io.c cmdline.c common.c \
 20         disk_io.c fsys_ext2fs.c fsys_fat.c fsys_ffs.c fsys_iso9660.c \
 21         fsys_jfs.c fsys_minix.c fsys_reiserfs.c fsys_ufs.c fsys_ufs2.c \
 22         fsys_zfs.c zfs_fletcher.c zfs_lzjb.c zfs_lz4.c zfs_sha256.c \
 22         fsys_zfs.c zfs_fletcher.c zfs_lzjb.c zfs_sha256.c \
 23         fsys_vstafs.c fsys_xfs.c gunzip.c md5.c serial.c stage2.c \
 24         terminfo.c tparm.c graphics.c
 25 libgrub_a_CFLAGS = $(GRUB_CFLAGS) -I$(top_srcdir)/lib \
 26         -DGRUB_UTIL=1 -DFSYS_EXT2FS=1 -DFSYS_FAT=1 -DFSYS_FFS=1 \
 27         -DFSYS_ISO9660=1 -DFSYS_JFS=1 -DFSYS_MINIX=1 -DFSYS_REISERFS=1 \
 28         -DFSYS_UFS=1 -DFSYS_UFS2=1 -DFSYS_ZFS=1 -DFSYS_VSTAFS=1 -DFSYS_XFS=1 \
 29         -DUSE_MD5_PASSWORDS=1 -DSUPPORT_SERIAL=1 -DSUPPORT_HERCULES=1 \

 31 # Stage 2 and Stage 1.5's.
 32 pkglibdir = $(libdir)/$(PACKAGE)/$(host_cpu)-$(host_vendor)

 34 EXTRA_PROGRAMS = nbloader.exec pxeloader.exec diskless.exec

 36 if DISKLESS_SUPPORT
 37 pkglib_DATA = stage2 stage2_eltorito e2fs_stage1_5 fat_stage1_5 \
 38         ffs_stage1_5 iso9660_stage1_5 jfs_stage1_5 minix_stage1_5 \
 39         reiserfs_stage1_5 ufs_stage1_5 ufs2_stage1_5 zfs_stage1_5 vstafs_stage1_
 40         xfs_stage1_5 nbgrub pxegrub
 41 noinst_DATA = pre_stage2 start start_eltorito nbloader pxeloader diskless
 42 noinst_PROGRAMS = pre_stage2.exec start.exec start_eltorito.exec \
 43         e2fs_stage1_5.exec fat_stage1_5.exec ffs_stage1_5.exec \
 44         iso9660_stage1_5.exec jfs_stage1_5.exec minix_stage1_5.exec \
 45         reiserfs_stage1_5.exec ufs_stage1_5.exec ufs2_stage1_5.exec \
 46         zfs_stage1_5.exec \
 47         vstafs_stage1_5.exec xfs_stage1_5.exec \
 48         nbloader.exec pxeloader.exec diskless.exec
 49 else
 50 pkglib_DATA = stage2 stage2_eltorito e2fs_stage1_5 fat_stage1_5 \
 51         ffs_stage1_5 iso9660_stage1_5 jfs_stage1_5 minix_stage1_5 \
 52         reiserfs_stage1_5 ufs_stage1_5 ufs2_stage1_5 zfs_stage1_5 vstafs_stage1_
 53         xfs_stage1_5
 54 noinst_DATA = pre_stage2 start start_eltorito
 55 noinst_PROGRAMS = pre_stage2.exec start.exec start_eltorito.exec \
 56         e2fs_stage1_5.exec fat_stage1_5.exec ffs_stage1_5.exec \
 57         iso9660_stage1_5.exec jfs_stage1_5.exec minix_stage1_5.exec \
 58         reiserfs_stage1_5.exec ufs_stage1_5.exec ufs2_stage1_5.exec \
 59         zfs_stage1_5.exec vstafs_stage1_5.exec xfs_stage1_5.exec
 60 endif
```



```
 61 MOSTLYCLEANFILES = $(noinst_PROGRAMS)

 63 PRE_STAGE2_LINK = -nostdlib -Wl,-N -Wl,-Ttext -Wl,8200
 64 START_LINK = -nostdlib -Wl,-N -Wl,-Ttext -Wl,8000
 65 NBLOADER_LINK = -nostdlib -Wl,-N -Wl,-Ttext -Wl,0
 66 PXELOADER_LINK = -nostdlib -Wl,-N -Wl,-Ttext -Wl,7C00
 67 START_ELTORITO_LINK = -nostdlib -Wl,-N -Wl,-Ttext -Wl,7C00

 69 if NETBOOT_SUPPORT
 70 NETBOOT_FLAGS = -I$(top_srcdir)/netboot -DSUPPORT_NETBOOT=1
 71 else
 72 NETBOOT_FLAGS =
 73 endif

 75 if SERIAL_SUPPORT
 76 SERIAL_FLAGS = -DSUPPORT_SERIAL=1
 77 else
 78 SERIAL_FLAGS =
 79 endif

 81 if HERCULES_SUPPORT
 82 HERCULES_FLAGS = -DSUPPORT_HERCULES=1
 83 else
 84 HERCULES_FLAGS =
 85 endif

 87 if GRAPHICS_SUPPORT
 88 GRAPHICS_FLAGS = -DSUPPORT_GRAPHICS=1
 89 else
 90 GRAPHICS_FLAGS =
 91 endif

 93 STAGE2_COMPILE = $(STAGE2_CFLAGS) -fno-builtin -nostdinc \
 94         $(NETBOOT_FLAGS) $(SERIAL_FLAGS) $(HERCULES_FLAGS) $(GRAPHICS_FLAGS)

 96 STAGE1_5_LINK = -nostdlib -Wl,-N -Wl,-Ttext -Wl,2000
 97 STAGE1_5_COMPILE = $(STAGE2_COMPILE) -DNO_DECOMPRESSION=1 -DSTAGE1_5=1

 99 # For stage2 target.
100 pre_stage2_exec_SOURCES = asm.S bios.c boot.c builtins.c char_io.c \
101         cmdline.c common.c console.c disk_io.c fsys_ext2fs.c \
102         fsys_fat.c fsys_ffs.c fsys_iso9660.c fsys_jfs.c fsys_minix.c \
103         fsys_reiserfs.c fsys_ufs.c fsys_ufs2.c fsys_vstafs.c fsys_xfs.c \
104         fsys_zfs.c zfs_fletcher.c zfs_lzjb.c zfs_lz4.c zfs_sha256.c \
104         fsys_zfs.c zfs_fletcher.c zfs_lzjb.c zfs_sha256.c \
105         gunzip.c hercules.c md5.c serial.c smp-imps.c stage2.c terminfo.c \
106         tparm.c graphics.c
107 pre_stage2_exec_CFLAGS = $(STAGE2_COMPILE) $(FSYS_CFLAGS)
108 pre_stage2_exec_CCASFLAGS = $(STAGE2_COMPILE) $(FSYS_CFLAGS)
109 pre_stage2_exec_LDFLAGS = $(PRE_STAGE2_LINK)

111 if NETBOOT_SUPPORT
112 pre_stage2_exec_LDADD = ../netboot/libdrivers.a
113 endif

115 if DISKLESS_SUPPORT
116 BUILT_SOURCES = stage2_size.h diskless_size.h
117 else
118 BUILT_SOURCES = stage2_size.h
119 endif

121 CLEANFILES = $(pkglib_DATA) $(noinst_DATA) $(BUILT_SOURCES)

123 stage2_size.h: pre_stage2
124         -rm -f stage2_size.h
125         set dummy 'ls -l pre_stage2'; \
```

```
126         echo "#define STAGE2_SIZE $$6" > stage2_size.h

128 start_exec_SOURCES = start.S
129 start_exec_CCASFLAGS = $(STAGE2_COMPILE)
130 start_exec_LDFLAGS = $(START_LINK)

132 # XXX: automake doesn't provide a way to specify dependencies for object
133 # files explicitly, so we must write this by a general Makefile scheme.
134 # If automake change the naming scheme for per-executable objects, this
135 # will be broken.
136 start_exec-start.$(OBJEXT): stage2_size.h

138 stage2: pre_stage2 start
139         -rm -f stage2
140         cat start pre_stage2 > stage2

142 start_eltorito_exec_SOURCES = start_eltorito.S
143 start_eltorito_exec_CCASFLAGS = $(STAGE2_COMPILE)
144 start_eltorito_exec_LDFLAGS = $(START_ELTORITO_LINK)

146 start_eltorito_exec-start.$(OBJEXT): stage2_size.h

148 stage2_eltorito: pre_stage2 start_eltorito
149         -rm -f stage2_eltorito
150         cat start_eltorito pre_stage2 > stage2_eltorito

152 # For e2fs_stage1_5 target.
153 e2fs_stage1_5_exec_SOURCES = start.S asm.S common.c char_io.c disk_io.c \
154         stage1_5.c fsys_ext2fs.c bios.c
155 e2fs_stage1_5_exec_CFLAGS = $(STAGE1_5_COMPILE) -DFSYS_EXT2FS=1 \
156         -DNO_BLOCK_FILES=1
157 e2fs_stage1_5_exec_CCASFLAGS = $(STAGE1_5_COMPILE) -DFSYS_EXT2FS=1 \
158         -DNO_BLOCK_FILES=1
159 e2fs_stage1_5_exec_LDFLAGS = $(STAGE1_5_LINK)

161 # For fat_stage1_5 target.
162 fat_stage1_5_exec_SOURCES = start.S asm.S common.c char_io.c disk_io.c \
163         stage1_5.c fsys_fat.c bios.c
164 fat_stage1_5_exec_CFLAGS = $(STAGE1_5_COMPILE) -DFSYS_FAT=1 \
165         -DNO_BLOCK_FILES=1
166 fat_stage1_5_exec_CCASFLAGS = $(STAGE1_5_COMPILE) -DFSYS_FAT=1 \
167         -DNO_BLOCK_FILES=1
168 fat_stage1_5_exec_LDFLAGS = $(STAGE1_5_LINK)

170 # For ffs_stage1_5 target.
171 ffs_stage1_5_exec_SOURCES = start.S asm.S common.c char_io.c disk_io.c \
172         stage1_5.c fsys_ffs.c bios.c
173 ffs_stage1_5_exec_CFLAGS = $(STAGE1_5_COMPILE) -DFSYS_FFS=1 \
174         -DNO_BLOCK_FILES=1
175 ffs_stage1_5_exec_CCASFLAGS = $(STAGE1_5_COMPILE) -DFSYS_FFS=1 \
176         -DNO_BLOCK_FILES=1
177 ffs_stage1_5_exec_LDFLAGS = $(STAGE1_5_LINK)

179 # For ufs_stage1_5 target.
180 ufs_stage1_5_exec_SOURCES = start.S asm.S common.c char_io.c disk_io.c \
181         stage1_5.c fsys_ufs.c bios.c
182 ufs_stage1_5_exec_CFLAGS = $(STAGE1_5_COMPILE) -DFSYS_UFS=1 \
183         -DNO_BLOCK_FILES=1
184 ufs_stage1_5_exec_CCASFLAGS = $(STAGE1_5_COMPILE) -DFSYS_UFS=1 \
185         -DNO_BLOCK_FILES=1
186 ufs_stage1_5_exec_LDFLAGS = $(STAGE1_5_LINK)

188 # For ufs2_stage1_5 target.
189 ufs2_stage1_5_exec_SOURCES = start.S asm.S common.c char_io.c disk_io.c \
190         stage1_5.c fsys_ufs2.c bios.c
191 ufs2_stage1_5_exec_CFLAGS = $(STAGE1_5_COMPILE) -DFSYS_UFS2=1 \
```



```
192         -DNO_BLOCK_FILES=1
193 ufs2_stage1_5_exec_CCASFLAGS = $(STAGE1_5_COMPILE) -DFSYS_UFS2=1 \
194         -DNO_BLOCK_FILES=1
195 ufs2_stage1_5_exec_LDFLAGS = $(STAGE1_5_LINK)

197 # For zfs_stage1_5 target.
198 zfs_stage1_5_exec_SOURCES = start.S asm.S common.c char_io.c disk_io.c \
199         stage1_5.c fsys_zfs.c zfs_fletcher.c zfs_lzjb.c zfs_lz4.c \
200         zfs_sha256.c bios.c
199         stage1_5.c fsys_zfs.c zfs_fletcher.c zfs_lzjb.c zfs_sha256.c bios.c
201 zfs_stage1_5_exec_CFLAGS = $(STAGE1_5_COMPILE) -DFSYS_ZFS=1 \
202         -DNO_BLOCK_FILES=1
203 zfs_stage1_5_exec_CCASFLAGS = $(STAGE1_5_COMPILE) -DFSYS_ZFS=1 \
204         -DNO_BLOCK_FILES=1
205 zfs_stage1_5_exec_LDFLAGS = $(STAGE1_5_LINK)

207 # For minix_stage1_5 target.
208 minix_stage1_5_exec_SOURCES = start.S asm.S common.c char_io.c disk_io.c \
209         stage1_5.c fsys_minix.c bios.c
210 minix_stage1_5_exec_CFLAGS = $(STAGE1_5_COMPILE) -DFSYS_MINIX=1 \
211         -DNO_BLOCK_FILES=1
212 minix_stage1_5_exec_CCASFLAGS = $(STAGE1_5_COMPILE) -DFSYS_MINIX=1 \
213         -DNO_BLOCK_FILES=1
214 minix_stage1_5_exec_LDFLAGS = $(STAGE1_5_LINK)

216 # For reiserfs_stage1_5 target.
217 reiserfs_stage1_5_exec_SOURCES = start.S asm.S common.c char_io.c \
218         disk_io.c stage1_5.c fsys_reiserfs.c bios.c
219 reiserfs_stage1_5_exec_CFLAGS = $(STAGE1_5_COMPILE) -DFSYS_REISERFS=1 \
220         -DNO_BLOCK_FILES=1
221 reiserfs_stage1_5_exec_CCASFLAGS = $(STAGE1_5_COMPILE) -DFSYS_REISERFS=1 \
222         -DNO_BLOCK_FILES=1
223 reiserfs_stage1_5_exec_LDFLAGS = $(STAGE1_5_LINK)

225 # For vstafs_stage1_5 target.
226 vstafs_stage1_5_exec_SOURCES = start.S asm.S common.c char_io.c \
227         disk_io.c stage1_5.c fsys_vstafs.c bios.c
228 vstafs_stage1_5_exec_CFLAGS = $(STAGE1_5_COMPILE) -DFSYS_VSTAFS=1 \
229         -DNO_BLOCK_FILES=1
230 vstafs_stage1_5_exec_CCASFLAGS = $(STAGE1_5_COMPILE) -DFSYS_VSTAFS=1 \
231         -DNO_BLOCK_FILES=1
232 vstafs_stage1_5_exec_LDFLAGS = $(STAGE1_5_LINK)

234 # For jfs_stage1_5 target.
235 jfs_stage1_5_exec_SOURCES = start.S asm.S common.c char_io.c \
236         disk_io.c stage1_5.c fsys_jfs.c bios.c
237 jfs_stage1_5_exec_CFLAGS = $(STAGE1_5_COMPILE) -DFSYS_JFS=1 \
238         -DNO_BLOCK_FILES=1
239 jfs_stage1_5_exec_CCASFLAGS = $(STAGE1_5_COMPILE) -DFSYS_JFS=1 \
240         -DNO_BLOCK_FILES=1
241 jfs_stage1_5_exec_LDFLAGS = $(STAGE1_5_LINK)

243 # For xfs_stage1_5 target.
244 xfs_stage1_5_exec_SOURCES = start.S asm.S common.c char_io.c \
245         disk_io.c stage1_5.c fsys_xfs.c bios.c
246 xfs_stage1_5_exec_CFLAGS = $(STAGE1_5_COMPILE) -DFSYS_XFS=1 \
247         -DNO_BLOCK_FILES=1
248 xfs_stage1_5_exec_CCASFLAGS = $(STAGE1_5_COMPILE) -DFSYS_XFS=1 \
249         -DNO_BLOCK_FILES=1
250 xfs_stage1_5_exec_LDFLAGS = $(STAGE1_5_LINK)

252 # For iso9660_stage1_5 target.
253 iso9660_stage1_5_exec_SOURCES = start_eltorito.S asm.S common.c char_io.c \
254         disk_io.c stage1_5.c fsys_iso9660.c bios.c
255 iso9660_stage1_5_exec_CFLAGS = $(STAGE1_5_COMPILE) -DFSYS_ISO9660=1 \
256         -DNO_BLOCK_FILES=1
```

```
257 iso9660_stage1_5_exec_CCASFLAGS = $(STAGE1_5_COMPILE) -DFSYS_ISO9660=1 \
258         -DNO_BLOCK_FILES=1
259 iso9660_stage1_5_exec_LDFLAGS = $(STAGE1_5_LINK)

261 # For diskless target.
262 diskless_exec_SOURCES = $(pre_stage2_exec_SOURCES)
263 diskless_exec_CFLAGS = $(STAGE2_COMPILE) $(FSYS_CFLAGS) \
264         -DSUPPORT_DISKLESS=1
265 diskless_exec_CCASFLAGS = $(STAGE2_COMPILE) $(FSYS_CFLAGS) \
266         -DSUPPORT_DISKLESS=1
267 diskless_exec_LDFLAGS = $(PRE_STAGE2_LINK)
268 diskless_exec_LDADD = ../netboot/libdrivers.a

270 diskless_size.h: diskless
271         -rm -f $@
272         set dummy `ls -l $^`; \
273         echo "#define DISKLESS_SIZE $$6" > $@

275 # For nbloader target.
276 nbloader_exec_SOURCES = nbloader.S
277 nbloader_exec_CCASFLAGS = $(STAGE2_COMPILE)
278 nbloader_exec_LDFLAGS = $(NBLOADER_LINK)

280 # XXX: See the comment for start_exec-start.o.
281 nbloader_exec-nbloader.$(OBJEXT): diskless_size.h

283 # For nbgrub target.
284 nbgrub: nbloader diskless
285         -rm -f $@
286         cat $^ > $@

288 # For pxeloader target.
289 pxeloader_exec_SOURCES = pxeloader.S
290 pxeloader_exec_CCASFLAGS = $(STAGE2_COMPILE)
291 pxeloader_exec_LDFLAGS = $(PXELOADER_LINK)

293 # XXX: See the comment for start_exec-start.o.
294 pxeloader_exec-pxeloader.$(OBJEXT): diskless_size.h

296 # For pxegrub target.
297 pxegrub: pxeloader diskless
298         -rm -f $@
299         cat $^ > $@

301 # General rule for making a raw binary.
302 SUFFIXES = .exec
303 .exec:
304         $(OBJCOPY) -O binary $< $@
```

```
**********************************************************
   27128 Tue Jan 15 15:38:21 2013
new/usr/src/grub/grub-0.97/stage2/Makefile.solaris
%B
**********************************************************
```

```
#
# Copyright 2005 Sun Microsystems, Inc.  All rights reserved.
# Use is subject to license terms.
# Copyright (c) 2013 by Saso Kiselkov. All rights reserved.
#
# ident "%Z%%M% %I%     %E% SMI"
#
include ../../Makefile.grub
include ../Makefile.solaris.defs

.SUFFIXES:
.SUFFIXES: .exec $(SUFFIXES)

.exec:
        $(OBJCOPY) -O binary $< $@


INCLUDES        = -I. -I.. -I../stage1 -I../netboot
LIBS            =

FSYS_CFLAGS     = -DFSYS_EXT2FS=1 -DFSYS_FAT=1 -DFSYS_FFS=1 -DFSYS_UFS=1 \
                  -DFSYS_ZFS=1 \
                  -DFSYS_UFS2=1 -DFSYS_REISERFS=1 -DFSYS_ISO9660=1 \
                  -DUSE_MD5_PASSWORDS=1
$(STAGE2_NETBOOT)TFTP_CFLAGS    = -DFSYS_TFTP=1
$(OPTION_FS)FSYS_CFLAGS += -DFSYS_MINIX=1 -DFSYS_JFS=1 -DFSYS_XFS=1 \
                  -DSYS_VSTAFS=1

GRAPHICS_CFLAGS = -DSUPPORT_GRAPHICS=1
# HERCULES_CFLAGS       = -DSUPPORT_HERCULES=1
NETBOOT_CFLAGS  = -DSUPPORT_NETBOOT=1
SERIAL_CFLAGS   = -DSUPPORT_SERIAL=1
GRUB_CFLAGS     = -O2 -I../lib -DGRUB_UTIL=1 \
                  -DFSYS_EXT2FS=1 -DFSYS_FAT=1 -DFSYS_FFS=1 -DFSYS_ISO9660=1 \
                  -DFSYS_ZFS=1 \
                  -DFSYS_REISERFS=1 -DFSYS_UFS=1 -DFSYS_UFS2=1 \
                  -DUSE_MD5_PASSWORDS=1 -DSUPPORT_SERIAL=1
$(OPTION_FS)GRUB_CFLAGS += -DFSYS_MINIX=1 -DFSYS_JFS=1 -DFSYS_XFS=1 \
                  -DSYS_VSTAFS=1
# GRUB_CFLAGS   += -DSUPPORT_HERCULES=1
STAGE1_5_CFLAGS = $(STAGE2_CFLAGS) -DNO_DECOMPRESSION=1 -DSTAGE1_5=1
STAGE2_CFLAGS   = -Os -fno-builtin -nostdinc \
                  $(SERIAL_CFLAGS) $(GRAPHICS_CFLAGS)
$(STAGE2_NETBOOT)STAGE2_CFLAGS  += $(NETBOOT_CFLAGS)
#STAGE2_CFLAGS  += $(HERCULES_CFLAGS)

NBLOADER_LINK   = -nostdlib -N -Ttext 0
PRE_STAGE2_LINK = -nostdlib -N -Ttext 8200
PXELOADER_LINK  = -nostdlib -N -Ttext 7C00
STAGE1_5_LINK   = -nostdlib -N -Ttext 2000
START_ELTORITO_LINK = -nostdlib -N -Ttext 7C00
START_LINK      = -nostdlib -N -Ttext 8000

LIBDRIVERS      = ../netboot/libdrivers.a


BUILT_SOURCES   = stage2_size.h
BUILT_SOURCES   += diskless_size.h


LIBRARIES       = $(LIBGRUB)
```



```
LIBGRUB         = libgrub.a
LIBGRUB_OBJS    = libgrub_a-boot.o              \
                  libgrub_a-builtins.o          \
                  libgrub_a-char_io.o           \
                  libgrub_a-cmdline.o           \
                  libgrub_a-common.o            \
                  libgrub_a-disk_io.o           \
                  libgrub_a-fsys_ext2fs.o       \
                  libgrub_a-fsys_fat.o          \
                  libgrub_a-fsys_ffs.o          \
                  libgrub_a-fsys_iso9660.o      \
                  libgrub_a-fsys_jfs.o          \
                  libgrub_a-fsys_minix.o        \
                  libgrub_a-fsys_reiserfs.o     \
                  libgrub_a-zfs_sha256.o        \
                  libgrub_a-zfs_lzjb.o          \
                  libgrub_a-zfs_lz4.o           \
                  libgrub_a-zfs_fletcher.o      \
                  libgrub_a-fsys_zfs.o          \
                  libgrub_a-fsys_ufs.o          \
                  libgrub_a-fsys_ufs2.o         \
                  libgrub_a-fsys_vstafs.o       \
                  libgrub_a-fsys_xfs.o          \
                  libgrub_a-graphics.o          \
                  libgrub_a-gunzip.o            \
                  libgrub_a-md5.o               \
                  libgrub_a-serial.o            \
                  libgrub_a-stage2.o            \
                  libgrub_a-terminfo.o          \
                  libgrub_a-tparm.o
$(LIBGRUB_OBJS) := CFLAGS = $(BASE_CFLAGS) $(GRUB_CFLAGS)


# Will be installed in $(ROOT_BOOT_GRUB).
INSTALL_DATA    = $(E2FS_STAGE1_5_DATA)         \
                  $(FAT_STAGE1_5_DATA)          \
                  $(FFS_STAGE1_5_DATA)          \
                  $(ISO9660_STAGE1_5_DATA)      \
                  $(JFS_STAGE1_5_DATA)          \
                  $(MINIX_STAGE1_5_DATA)        \
                  $(NBGRUB_DATA)                \
                  $(PXEGRUB_DATA)               \
                  $(REISERFS_STAGE1_5_DATA)     \
                  $(STAGE2_DATA)                \
                  $(STAGE2_ELTORITO_DATA)       \
                  $(UFS2_STAGE1_5_DATA)         \
                  $(UFS_STAGE1_5_DATA)          \
                  $(ZFS_STAGE1_5_DATA)          \
                  $(VSTAFS_STAGE1_5_DATA)       \
                  $(XFS_STAGE1_5_DATA)

# Won't be installed.
NOINSTALL_DATA  = $(DISKLESS_DATA)              \
                  $(NBLOADER_DATA)              \
                  $(PRE_STAGE2_DATA)            \
                  $(PXELOADER_DATA)             \
                  $(START_DATA)                 \
                  $(START_ELTORITO_DATA)

DATA            = $(INSTALL_DATA) $(NOINSTALL_DATA)


PROGRAMS        = $(DISKLESS_EXEC)              \
                  $(E2FS_STAGE1_5_EXEC)         \
                  $(FAT_STAGE1_5_EXEC)          \
```

```
128                     $(FFS_STAGE1_5_EXEC)            \
129                     $(ISO9660_STAGE1_5_EXEC)        \
130                     $(JFS_STAGE1_5_EXEC)            \
131                     $(MINIX_STAGE1_5_EXEC)          \
132                     $(NBLOADER_EXEC)                \
133                     $(PRE_STAGE2_EXEC)              \
134                     $(PXELOADER_EXEC)               \
135                     $(REISERFS_STAGE1_5_EXEC)       \
136                     $(START_EXEC)                   \
137                     $(START_ELTORITO_EXEC)          \
138                     $(UFS2_STAGE1_5_EXEC)           \
139                     $(UFS_STAGE1_5_EXEC)            \
140                     $(ZFS_STAGE1_5_EXEC)            \
141                     $(VSTAFS_STAGE1_5_EXEC)         \
142                     $(XFS_STAGE1_5_EXEC)


145 #
146 # diskless
147 #
148 DISKLESS_DATA           = diskless
149 DISKLESS_EXEC           = diskless.exec
150 DISKLESS_ASMOBJS        = diskless_exec-asm.o

152 DISKLESS_OBJS           = diskless_exec-bios.o          \
153                           diskless_exec-boot.o          \
154                           diskless_exec-builtins.o      \
155                           diskless_exec-char_io.o       \
156                           diskless_exec-cmdline.o       \
157                           diskless_exec-common.o        \
158                           diskless_exec-console.o       \
159                           diskless_exec-disk_io.o       \
160                           diskless_exec-fsys_ext2fs.o   \
161                           diskless_exec-fsys_fat.o      \
162                           diskless_exec-fsys_ffs.o      \
163                           diskless_exec-fsys_iso9660.o  \
164                           diskless_exec-fsys_jfs.o      \
165                           diskless_exec-fsys_minix.o    \
166                           diskless_exec-fsys_reiserfs.o \
167                           diskless_exec-zfs_sha256.o    \
168                           diskless_exec-zfs_lzjb.o      \
169                           diskless_exec-zfs_lz4.o       \
170                           diskless_exec-zfs_fletcher.o  \
171                           diskless_exec-fsys_zfs.o      \
172                           diskless_exec-fsys_ufs.o      \
173                           diskless_exec-fsys_ufs2.o     \
174                           diskless_exec-fsys_vstafs.o   \
175                           diskless_exec-fsys_xfs.o      \
176                           diskless_exec-graphics.o      \
177                           diskless_exec-gunzip.o        \
178                           diskless_exec-hercules.o      \
179                           diskless_exec-md5.o           \
180                           diskless_exec-serial.o        \
181                           diskless_exec-smp-imps.o      \
182                           diskless_exec-stage2.o        \
183                           diskless_exec-terminfo.o      \
184                           diskless_exec-tparm.o

186 $(DISKLESS_EXEC)         := LDFLAGS = $(BASE_LDFLAGS) $(PRE_STAGE2_LINK)
187 $(DISKLESS_EXEC)         := LIBS = $(LIBDRIVERS)

189 $(DISKLESS_ASMOBJS)      := CCASFLAGS = $(BASE_CCASFLAGS) $(STAGE2_CFLAGS) \
190                                 $(FSYS_CFLAGS) -DFSYS_TFTP=1 \
191                                 -DSUPPORT_DISKLESS=1

193 $(DISKLESS_OBJS)         := CFLAGS = $(BASE_CFLAGS) $(STAGE2_CFLAGS) \
```



```
194                                 $(FSYS_CFLAGS) -DFSYS_TFTP=1 \
195                                 -DSUPPORT_DISKLESS=1

197 #
198 # ext2fs stage 1.5
199 #
200 E2FS_STAGE1_5_DATA      = e2fs_stage1_5
201 E2FS_STAGE1_5_EXEC      = e2fs_stage1_5.exec

203 E2FS_STAGE1_5_ASMOBJS   = e2fs_stage1_5_exec-asm.o \
204                           e2fs_stage1_5_exec-start.o

206 E2FS_STAGE1_5_OBJS      = e2fs_stage1_5_exec-bios.o \
207                           e2fs_stage1_5_exec-char_io.o \
208                           e2fs_stage1_5_exec-common.o \
209                           e2fs_stage1_5_exec-disk_io.o \
210                           e2fs_stage1_5_exec-fsys_ext2fs.o \
211                           e2fs_stage1_5_exec-stage1_5.o

213 $(E2FS_STAGE1_5_EXEC)   := LDFLAGS = $(BASE_LDFLAGS) $(STAGE1_5_LINK)

215 $(E2FS_STAGE1_5_ASMOBJS) := CCASFLAGS = $(BASE_CCASFLAGS) $(STAGE1_5_CFLAGS) \
216                                 -DFSYS_EXT2FS=1 -DNO_BLOCK_FILES=1

218 $(E2FS_STAGE1_5_OBJS)   := CFLAGS = $(BASE_CFLAGS) $(STAGE1_5_CFLAGS) \
219                                 -DFSYS_EXT2FS=1 -DNO_BLOCK_FILES=1

221 #
222 # FAT stage 1.5
223 #
224 FAT_STAGE1_5_DATA       = fat_stage1_5
225 FAT_STAGE1_5_EXEC       = fat_stage1_5.exec

227 FAT_STAGE1_5_ASMOBJS    = fat_stage1_5_exec-asm.o \
228                           fat_stage1_5_exec-start.o

230 FAT_STAGE1_5_OBJS       = fat_stage1_5_exec-bios.o \
231                           fat_stage1_5_exec-char_io.o \
232                           fat_stage1_5_exec-common.o \
233                           fat_stage1_5_exec-disk_io.o \
234                           fat_stage1_5_exec-fsys_fat.o \
235                           fat_stage1_5_exec-stage1_5.o

237 $(FAT_STAGE1_5_EXEC)    := LDFLAGS = $(BASE_LDFLAGS) $(STAGE1_5_LINK)

239 $(FAT_STAGE1_5_ASMOBJS) := CCASFLAGS = $(BASE_CCASFLAGS) $(STAGE1_5_CFLAGS) \
240                                 -DFSYS_FAT=1 -DNO_BLOCK_FILES=1

242 $(FAT_STAGE1_5_OBJS)    := CFLAGS = $(BASE_CFLAGS) $(STAGE1_5_CFLAGS) \
243                                 -DFSYS_FAT=1 -DNO_BLOCK_FILES=1

245 #
246 # FFS stage 1.5
247 #
248 FFS_STAGE1_5_DATA       = ffs_stage1_5
249 FFS_STAGE1_5_EXEC       = ffs_stage1_5.exec

251 FFS_STAGE1_5_ASMOBJS    = ffs_stage1_5_exec-asm.o \
252                           ffs_stage1_5_exec-start.o

254 FFS_STAGE1_5_OBJS       = ffs_stage1_5_exec-bios.o \
255                           ffs_stage1_5_exec-char_io.o \
256                           ffs_stage1_5_exec-common.o \
257                           ffs_stage1_5_exec-disk_io.o \
258                           ffs_stage1_5_exec-fsys_ffs.o \
259                           ffs_stage1_5_exec-stage1_5.o
```

```
261 $(FFS_STAGE1_5_EXEC)    := LDFLAGS = $(BASE_LDFLAGS) $(STAGE1_5_LINK)

263 $(FFS_STAGE1_5_ASMOBJS) := CCASFLAGS = $(BASE_CCASFLAGS) $(STAGE1_5_CFLAGS) \
264                               -DFSYS_FFS=1 -DNO_BLOCK_FILES=1

266 $(FFS_STAGE1_5_OBJS)    := CFLAGS = $(BASE_CFLAGS) $(STAGE1_5_CFLAGS) \
267                               -DFSYS_FFS=1 -DNO_BLOCK_FILES=1

269 #
270 # ISO9660 stage 1.5
271 #
272 ISO9660_STAGE1_5_DATA   = iso9660_stage1_5
273 ISO9660_STAGE1_5_EXEC   = iso9660_stage1_5.exec

275 ISO9660_STAGE1_5_ASMOBJS = iso9660_stage1_5_exec-asm.o \
276                            iso9660_stage1_5_exec-start_eltorito.o

278 ISO9660_STAGE1_5_OBJS   = iso9660_stage1_5_exec-bios.o \
279                           iso9660_stage1_5_exec-char_io.o \
280                           iso9660_stage1_5_exec-common.o \
281                           iso9660_stage1_5_exec-disk_io.o \
282                           iso9660_stage1_5_exec-fsys_iso9660.o \
283                           iso9660_stage1_5_exec-stage1_5.o

285 $(ISO9660_STAGE1_5_EXEC) := LDFLAGS = $(BASE_LDFLAGS) $(STAGE1_5_LINK)

287 $(ISO9660_STAGE1_5_ASMOBJS) := CCASFLAGS = $(BASE_CCASFLAGS) \
288                               $(STAGE1_5_CFLAGS) \
289                               -DFSYS_ISO9660=1 -DNO_BLOCK_FILES=1

291 $(ISO9660_STAGE1_5_OBJS) := CFLAGS = $(BASE_CFLAGS) $(STAGE1_5_CFLAGS) \
292                               -DFSYS_ISO9660=1 -DNO_BLOCK_FILES=1

294 #
295 # IBM JFS stage 1.5
296 #
297 JFS_STAGE1_5_DATA       = jfs_stage1_5
298 JFS_STAGE1_5_EXEC       = jfs_stage1_5.exec

300 JFS_STAGE1_5_ASMOBJS    = jfs_stage1_5_exec-asm.o \
301                           jfs_stage1_5_exec-start.o

303 JFS_STAGE1_5_OBJS       = jfs_stage1_5_exec-bios.o \
304                           jfs_stage1_5_exec-char_io.o \
305                           jfs_stage1_5_exec-common.o \
306                           jfs_stage1_5_exec-disk_io.o \
307                           jfs_stage1_5_exec-fsys_jfs.o \
308                           jfs_stage1_5_exec-stage1_5.o

310 $(JFS_STAGE1_5_EXEC)    := LDFLAGS = $(BASE_LDFLAGS) $(STAGE1_5_LINK)

312 $(JFS_STAGE1_5_ASMOBJS) := CCASFLAGS = $(BASE_CCASFLAGS) $(STAGE1_5_CFLAGS) \
313                               -DFSYS_JFS=1 -DNO_BLOCK_FILES=1

315 $(JFS_STAGE1_5_OBJS)    := CFLAGS = $(BASE_CFLAGS) $(STAGE1_5_CFLAGS) \
316                               -DFSYS_JFS=1 -DNO_BLOCK_FILES=1

318 #
319 # Minix fs stage 1.5
320 #
321 MINIX_STAGE1_5_DATA     = minix_stage1_5
322 MINIX_STAGE1_5_EXEC     = minix_stage1_5.exec

324 MINIX_STAGE1_5_ASMOBJS  = minix_stage1_5_exec-asm.o \
325                           minix_stage1_5_exec-start.o
```



```
327 MINIX_STAGE1_5_OBJS     = minix_stage1_5_exec-bios.o \
328                           minix_stage1_5_exec-char_io.o \
329                           minix_stage1_5_exec-common.o \
330                           minix_stage1_5_exec-disk_io.o \
331                           minix_stage1_5_exec-fsys_minix.o \
332                           minix_stage1_5_exec-stage1_5.o

334 $(MINIX_STAGE1_5_EXEC)  := LDFLAGS = $(BASE_LDFLAGS) $(STAGE1_5_LINK)

336 $(MINIX_STAGE1_5_ASMOBJS) := CCASFLAGS = $(BASE_CCASFLAGS) $(STAGE1_5_CFLAGS) \
337                               -DFSYS_MINIX=1 -DNO_BLOCK_FILES=1

339 $(MINIX_STAGE1_5_OBJS)  := CFLAGS = $(BASE_CFLAGS) $(STAGE1_5_CFLAGS) \
340                               -DFSYS_MINIX=1 -DNO_BLOCK_FILES=1

342 #
343 # nbgrub
344 #
345 NBGRUB_DATA             = nbgrub

347 #
348 # nbloader
349 #
350 NBLOADER_DATA           = nbloader
351 NBLOADER_EXEC           = nbloader.exec
352 NBLOADER_ASMOBJS        = nbloader_exec-nbloader.o
353 NBLOADER_OBJS           =
354 $(NBLOADER_EXEC)        := LDFLAGS = $(BASE_LDFLAGS) $(NBLOADER_LINK)
355 $(NBLOADER_ASMOBJS)     := CCASFLAGS = $(BASE_CCASFLAGS) $(STAGE2_CFLAGS)

357 #
358 # Pre-stage2
359 #
360 PRE_STAGE2_DATA         = pre_stage2
361 PRE_STAGE2_EXEC         = pre_stage2.exec
362 PRE_STAGE2_ASMOBJS      = pre_stage2_exec-asm.o

364 PRE_STAGE2_OBJS         = pre_stage2_exec-bios.o \
365                           pre_stage2_exec-boot.o \
366                           pre_stage2_exec-builtins.o \
367                           pre_stage2_exec-char_io.o \
368                           pre_stage2_exec-cmdline.o \
369                           pre_stage2_exec-common.o \
370                           pre_stage2_exec-console.o \
371                           pre_stage2_exec-disk_io.o \
372                           pre_stage2_exec-fsys_ext2fs.o \
373                           pre_stage2_exec-fsys_fat.o \
374                           pre_stage2_exec-fsys_ffs.o \
375                           pre_stage2_exec-fsys_iso9660.o \
376                           pre_stage2_exec-fsys_jfs.o \
377                           pre_stage2_exec-fsys_minix.o \
378                           pre_stage2_exec-fsys_reiserfs.o \
379                           pre_stage2_exec-zfs_sha256.o \
380                           pre_stage2_exec-zfs_lzjb.o \
381                           pre_stage2_exec-zfs_lz4.o \
382                           pre_stage2_exec-zfs_fletcher.o \
383                           pre_stage2_exec-fsys_zfs.o \
384                           pre_stage2_exec-fsys_ufs.o \
385                           pre_stage2_exec-fsys_ufs2.o \
386                           pre_stage2_exec-fsys_vstafs.o \
387                           pre_stage2_exec-fsys_xfs.o \
388                           pre_stage2_exec-graphics.o \
389                           pre_stage2_exec-gunzip.o \
390                           pre_stage2_exec-hercules.o \
391                           pre_stage2_exec-md5.o \
```

```
392                             pre_stage2_exec-serial.o \
393                             pre_stage2_exec-smp-imps.o \
394                             pre_stage2_exec-stage2.o \
395                             pre_stage2_exec-terminfo.o \
396                             pre_stage2_exec-tparm.o

398 $(PRE_STAGE2_EXEC)      := LDFLAGS = $(BASE_LDFLAGS) $(PRE_STAGE2_LINK)
399 $(STAGE2_NETBOOT)$(PRE_STAGE2_EXEC)     := LIBS = $(LIBDRIVERS)

401 $(PRE_STAGE2_ASMOBJS)   := CCASFLAGS = $(BASE_CCASFLAGS) $(STAGE2_CFLAGS) \
402                                 $(FSYS_CFLAGS) $(TFTP_CFLAGS)

404 $(PRE_STAGE2_OBJS)      := CFLAGS = $(BASE_CFLAGS) $(STAGE2_CFLAGS) \
405                                 $(FSYS_CFLAGS) $(TFTP_CFLAGS)

407 #
408 # pxegrub
409 #
410 PXEGRUB_DATA            = pxegrub

412 #
413 # pxeloader
414 #
415 PXELOADER_DATA          = pxeloader
416 PXELOADER_EXEC          = pxeloader.exec
417 PXELOADER_ASMOBJS       = pxeloader_exec-pxeloader.o
418 PXELOADER_OBJS          =
419 $(PXELOADER_EXEC)       := LDFLAGS = $(BASE_LDFLAGS) $(PXELOADER_LINK)
420 $(PXELOADER_ASMOBJS)    := CCASFLAGS = $(BASE_CCASFLAGS) $(STAGE2_CFLAGS)

422 #
423 # ReiserFS stage 1.5
424 #
425 REISERFS_STAGE1_5_DATA  = reiserfs_stage1_5
426 REISERFS_STAGE1_5_EXEC  = reiserfs_stage1_5.exec

428 REISERFS_STAGE1_5_ASMOBJS = reiserfs_stage1_5_exec-asm.o \
429                             reiserfs_stage1_5_exec-start.o

431 REISERFS_STAGE1_5_OBJS  = reiserfs_stage1_5_exec-bios.o \
432                           reiserfs_stage1_5_exec-char_io.o \
433                           reiserfs_stage1_5_exec-common.o \
434                           reiserfs_stage1_5_exec-disk_io.o \
435                           reiserfs_stage1_5_exec-fsys_reiserfs.o \
436                           reiserfs_stage1_5_exec-stage1_5.o
437
438 $(REISERFS_STAGE1_5_EXEC) := LDFLAGS = $(BASE_LDFLAGS) $(STAGE1_5_LINK)
439
440 $(REISERFS_STAGE1_5_ASMOBJS) := CCASFLAGS = $(BASE_CCASFLAGS) \
441                                 $(STAGE1_5_CFLAGS) \
442                                 -DFSYS_REISERFS=1 -DNO_BLOCK_FILES=1
443
444 $(REISERFS_STAGE1_5_OBJS) := CFLAGS = $(BASE_CFLAGS) $(STAGE1_5_CFLAGS) \
445                                 -DFSYS_REISERFS=1 -DNO_BLOCK_FILES=1

447 #
448 # stage2
449 #
450 STAGE2_DATA             = stage2

452 #
453 # Eltorito stage2
454 #
455 STAGE2_ELTORITO_DATA    = stage2_eltorito

457 #
```



```
458 # start
459 #
460 START_DATA              = start
461 START_EXEC              = start.exec
462 START_ASMOBJS           = start_exec-start.o
463 START_OBJS              =
464 $(START_EXEC)           := LDFLAGS = $(BASE_LDFLAGS) $(START_LINK)
465 $(START_ASMOBJS)        := CCASFLAGS = $(BASE_CCASFLAGS) $(STAGE2_CFLAGS)

467 #
468 # Eltorito start
469 #
470 START_ELTORITO_DATA     = start_eltorito
471 START_ELTORITO_EXEC     = start_eltorito.exec
472 START_ELTORITO_ASMOBJS  = start_eltorito_exec-start_eltorito.o
473 START_ELTORITO_OBJS     =
474 $(START_ELTORITO_EXEC)  := LDFLAGS = $(BASE_LDFLAGS) $(START_ELTORITO_LINK)
475 $(START_ELTORITO_ASMOBJS) := CCASFLAGS = $(BASE_CCASFLAGS) $(STAGE2_CFLAGS)

477 #
478 # UFS2 stage 1.5
479 #
480 UFS2_STAGE1_5_DATA      = ufs2_stage1_5
481 UFS2_STAGE1_5_EXEC      = ufs2_stage1_5.exec

483 UFS2_STAGE1_5_ASMOBJS   = ufs2_stage1_5_exec-asm.o \
484                           ufs2_stage1_5_exec-start.o

486 UFS2_STAGE1_5_OBJS      = ufs2_stage1_5_exec-bios.o \
487                           ufs2_stage1_5_exec-char_io.o \
488                           ufs2_stage1_5_exec-common.o \
489                           ufs2_stage1_5_exec-disk_io.o \
490                           ufs2_stage1_5_exec-fsys_ufs2.o \
491                           ufs2_stage1_5_exec-stage1_5.o

493 $(UFS2_STAGE1_5_EXEC)   := LDFLAGS = $(BASE_LDFLAGS) $(STAGE1_5_LINK)

495 $(UFS2_STAGE1_5_ASMOBJS) := CCASFLAGS = $(BASE_CCASFLAGS) $(STAGE1_5_CFLAGS) \
496                                 -DFSYS_UFS2=1 -DNO_BLOCK_FILES=1

498 $(UFS2_STAGE1_5_OBJS)   := CFLAGS = $(BASE_CFLAGS) $(STAGE1_5_CFLAGS) \
499                                 -DFSYS_UFS2=1 -DNO_BLOCK_FILES=1

501 #
502 # UFS stage 1.5
503 #
504 UFS_STAGE1_5_DATA       = ufs_stage1_5
505 UFS_STAGE1_5_EXEC       = ufs_stage1_5.exec

507 UFS_STAGE1_5_ASMOBJS    = ufs_stage1_5_exec-asm.o \
508                           ufs_stage1_5_exec-start.o

510 UFS_STAGE1_5_OBJS       = ufs_stage1_5_exec-bios.o \
511                           ufs_stage1_5_exec-char_io.o \
512                           ufs_stage1_5_exec-common.o \
513                           ufs_stage1_5_exec-disk_io.o \
514                           ufs_stage1_5_exec-fsys_ufs.o \
515                           ufs_stage1_5_exec-stage1_5.o

517 $(UFS_STAGE1_5_EXEC)    := LDFLAGS = $(BASE_LDFLAGS) $(STAGE1_5_LINK)

519 $(UFS_STAGE1_5_ASMOBJS) := CCASFLAGS = $(BASE_CCASFLAGS) $(STAGE1_5_CFLAGS) \
520                                 -DFSYS_UFS=1 -DNO_BLOCK_FILES=1

522 $(UFS_STAGE1_5_OBJS)    := CFLAGS = $(BASE_CFLAGS) $(STAGE1_5_CFLAGS) \
523                                 -DFSYS_UFS=1 -DNO_BLOCK_FILES=1
```

```
525 #
526 # ZFS stage 1.5
527 #
528 ZFS_STAGE1_5_DATA       = zfs_stage1_5
529 ZFS_STAGE1_5_EXEC       = zfs_stage1_5.exec

531 ZFS_STAGE1_5_ASMOBJS    = zfs_stage1_5_exec-asm.o \
532                           zfs_stage1_5_exec-start.o

534 ZFS_STAGE1_5_OBJS       = zfs_stage1_5_exec-bios.o \
535                           zfs_stage1_5_exec-char_io.o \
536                           zfs_stage1_5_exec-common.o \
537                           zfs_stage1_5_exec-disk_io.o \
538                           zfs_stage1_5_exec-zfs_sha256.o \
539                           zfs_stage1_5_exec-zfs_lzjb.o \
540                           zfs_stage1_5_exec-zfs_lz4.o \
541                           zfs_stage1_5_exec-zfs_fletcher.o \
542                           zfs_stage1_5_exec-fsys_zfs.o \
543                           zfs_stage1_5_exec-stage1_5.o

545 $(ZFS_STAGE1_5_EXEC)    := LDFLAGS = $(BASE_LDFLAGS) $(STAGE1_5_LINK)

547 $(ZFS_STAGE1_5_ASMOBJS) := CCASFLAGS = $(BASE_CCASFLAGS) $(STAGE1_5_CFLAGS) \
548                                -DFSYS_ZFS=1 -DNO_BLOCK_FILES=1

550 $(ZFS_STAGE1_5_OBJS)    := CFLAGS = $(BASE_CFLAGS) $(STAGE1_5_CFLAGS) \
551                                -DFSYS_ZFS=1 -DNO_BLOCK_FILES=1

553 #
554 # VSTa FS
555 #
556 VSTAFS_STAGE1_5_DATA    = vstafs_stage1_5
557 VSTAFS_STAGE1_5_EXEC    = vstafs_stage1_5.exec

559 VSTAFS_STAGE1_5_ASMOBJS = vstafs_stage1_5_exec-asm.o \
560                           vstafs_stage1_5_exec-start.o

562 VSTAFS_STAGE1_5_OBJS    = vstafs_stage1_5_exec-bios.o \
563                           vstafs_stage1_5_exec-char_io.o \
564                           vstafs_stage1_5_exec-common.o \
565                           vstafs_stage1_5_exec-disk_io.o \
566                           vstafs_stage1_5_exec-fsys_vstafs.o \
567                           vstafs_stage1_5_exec-stage1_5.o

569 $(VSTAFS_STAGE1_5_EXEC) := LDFLAGS = $(BASE_LDFLAGS) $(STAGE1_5_LINK)

571 $(VSTAFS_STAGE1_5_ASMOBJS) := CCASFLAGS = $(BASE_CCASFLAGS) $(STAGE1_5_CFLAGS) \
572                                -DFSYS_VSTAFS=1 -DNO_BLOCK_FILES=1

574 $(VSTAFS_STAGE1_5_OBJS) := CFLAGS = $(BASE_CFLAGS) $(STAGE1_5_CFLAGS) \
575                                -DFSYS_VSTAFS=1 -DNO_BLOCK_FILES=1

577 #
578 # SGI XFS stage 1.5
579 #
580 XFS_STAGE1_5_DATA       = xfs_stage1_5
581 XFS_STAGE1_5_EXEC       = xfs_stage1_5.exec

583 XFS_STAGE1_5_ASMOBJS    = xfs_stage1_5_exec-asm.o \
584                           xfs_stage1_5_exec-start.o

586 XFS_STAGE1_5_OBJS       = xfs_stage1_5_exec-bios.o \
587                           xfs_stage1_5_exec-char_io.o \
588                           xfs_stage1_5_exec-common.o \
589                           xfs_stage1_5_exec-disk_io.o \
```



```
590                           xfs_stage1_5_exec-fsys_xfs.o \
591                           xfs_stage1_5_exec-stage1_5.o

593 $(XFS_STAGE1_5_EXEC)    := LDFLAGS = $(BASE_LDFLAGS) $(STAGE1_5_LINK)

595 $(XFS_STAGE1_5_ASMOBJS) := CCASFLAGS = $(BASE_CCASFLAGS) $(STAGE1_5_CFLAGS) \
596                                -DFSYS_XFS=1 -DNO_BLOCK_FILES=1

598 $(XFS_STAGE1_5_OBJS)    := CFLAGS = $(BASE_CFLAGS) $(STAGE1_5_CFLAGS) \
599                                -DFSYS_XFS=1 -DNO_BLOCK_FILES=1


602 INSTALL_TARGETS = $(INSTALL_DATA:%=$(ROOT_BOOT_GRUB)/%) \
603                $(SRC_ZFSINC_DIR)

605 #
606 # Source files to be installed in $(ROOT_SRC).
607 #
608 SRC_DIR         = $(ROOT_SRC)/stage2
609 SRC_ZFSINC_DIR  = $(SRC_DIR)/zfs-include
610 SRC_FILES       = Makefile.am Makefile.in apic.h apm.S asm.S bios.c boot.c \
611                   builtins.c char_io.c cmdline.c common.c console.c defs.h \
612                   dir.h disk_inode.h disk_inode_ffs.h disk_io.c fat.h \
613                   filesys.h freebsd.h fs.h fsys_ext2fs.c fsys_fat.c \
614                   fsys_ffs.c fsys_iso9660.c fsys_jfs.c fsys_minix.c \
615                   zfs_sha256.c zfs_lzjb.c zfs_lz4.c zfs_fletcher.c fsys_zfs.c \
610                   zfs_sha256.c zfs_lzjb.c zfs_fletcher.c fsys_zfs.c \
616                   fsys_reiserfs.c fsys_ufs.c fsys_ufs2.c fsys_vstafs.c \
617                   fsys_xfs.c graphics.c graphics.h gunzip.c hercules.c \
618                   hercules.h i386-elf.h imgact_aout.h iso9660.h jfs.h \
619                   mb_header.h mb_info.h md5.c md5.h nbi.h nbloader.S \
620                   pc_slice.h pxeloader.S serial.c serial.h setjmp.S shared.h \
621                   size_test smp-imps.c smp-imps.h stage1_5.c stage2.c \
622                   start.S start_eltorito.S term.h terminfo.c terminfo.h \
623                   tparm.c tparm.h fsys_zfs.h ufs.h ufs2.h vstafs.h xfs.h
624 SRC_ZFSINC_FILES = dmu.h dmu_objset.h dnode.h dsl_dataset.h dsl_dir.h spa.h \
625                uberblock_impl.h vdev_impl.h zap_impl.h zap_leaf.h zfs.h \
626                zfs_acl.h zfs_znode.h zil.h zio.h zio_checksum.h

628 INSTALL_TARGETS += $(SRC_FILES:%=$(SRC_DIR)/%) \
629                $(SRC_ZFSINC_FILES:%=$(SRC_ZFSINC_DIR)/%)


632 ###############################################################################
633 all: $(BUILT_SOURCES) .WAIT $(LIBRARIES) $(PROGRAMS) $(DATA)

635 #
636 # BUILT_SOURCES
637 #
638 stage2_size.h: $(PRE_STAGE2_DATA)
639         $(RM) $@
640         set dummy `ls -l $(PRE_STAGE2_DATA)`; \
641         echo "#define STAGE2_SIZE $$6" > $@

643 diskless_size.h: $(DISKLESS_DATA)
644         $(RM) $@
645         set dummy `ls -l $(DISKLESS_DATA)`; \
646         echo "#define DISKLESS_SIZE $$6" > $@

648 #
649 # LIBRARIES
650 #
651 $(LIBGRUB): $(LIBGRUB_OBJS)
652         $(RM) $@
653         $(AR) cru $@ $(LIBGRUB_OBJS)
654         $(RANLIB) $@
```

```
656 $(LIBGRUB_OBJS): $$(@:libgrub_a-%.o=%.c)
657         $(CC) $(CFLAGS) -c -o $@ $(@:libgrub_a-%.o=%.c)


660 #
661 # Diskless
662 #
663 $(DISKLESS_EXEC): $(DISKLESS_ASMOBJS) $(DISKLESS_OBJS)
664         $(RM) $@
665         $(LINK) -o $@ $(DISKLESS_ASMOBJS) $(DISKLESS_OBJS) $(LIBS)

667 $(DISKLESS_ASMOBJS): $$(@:diskless_exec-%.o=%.S)
668         $(CCAS) $(CCASFLAGS) -c -o $@ $(@:diskless_exec-%.o=%.S)

670 $(DISKLESS_OBJS): $$(@:diskless_exec-%.o=%.c)
671         $(CC) $(CFLAGS) -c -o $@ $(@:diskless_exec-%.o=%.c)

673 #
674 # ext2fs stage 1.5
675 #
676 $(E2FS_STAGE1_5_EXEC): $(E2FS_STAGE1_5_ASMOBJS) $(E2FS_STAGE1_5_OBJS)
677         $(RM) $@
678         $(LINK) -o $@ $(E2FS_STAGE1_5_ASMOBJS) $(E2FS_STAGE1_5_OBJS) $(LIBS)

680 $(E2FS_STAGE1_5_ASMOBJS): $$(@:e2fs_stage1_5_exec-%.o=%.S)
681         $(CCAS) $(CCASFLAGS) -c -o $@ $(@:e2fs_stage1_5_exec-%.o=%.S)

683 $(E2FS_STAGE1_5_OBJS): $$(@:e2fs_stage1_5_exec-%.o=%.c)
684         $(CC) $(CFLAGS) -c -o $@ $(@:e2fs_stage1_5_exec-%.o=%.c)

686 #
687 # FAT stage 1.5
688 #
689 $(FAT_STAGE1_5_EXEC): $(FAT_STAGE1_5_ASMOBJS) $(FAT_STAGE1_5_OBJS)
690         $(RM) $@
691         $(LINK) -o $@ $(FAT_STAGE1_5_ASMOBJS) $(FAT_STAGE1_5_OBJS) $(LIBS)

693 $(FAT_STAGE1_5_ASMOBJS): $$(@:fat_stage1_5_exec-%.o=%.S)
694         $(CCAS) $(CCASFLAGS) -c -o $@ $(@:fat_stage1_5_exec-%.o=%.S)

696 $(FAT_STAGE1_5_OBJS): $$(@:fat_stage1_5_exec-%.o=%.c)
697         $(CC) $(CFLAGS) -c -o $@ $(@:fat_stage1_5_exec-%.o=%.c)

699 #
700 # FFS stage 1.5
701 #
702 $(FFS_STAGE1_5_EXEC): $(FFS_STAGE1_5_ASMOBJS) $(FFS_STAGE1_5_OBJS)
703         $(RM) $@
704         $(LINK) -o $@ $(FFS_STAGE1_5_ASMOBJS) $(FFS_STAGE1_5_OBJS) $(LIBS)

706 $(FFS_STAGE1_5_ASMOBJS): $$(@:ffs_stage1_5_exec-%.o=%.S)
707         $(CCAS) $(CCASFLAGS) -c -o $@ $(@:ffs_stage1_5_exec-%.o=%.S)

709 $(FFS_STAGE1_5_OBJS): $$(@:ffs_stage1_5_exec-%.o=%.c)
710         $(CC) $(CFLAGS) -c -o $@ $(@:ffs_stage1_5_exec-%.o=%.c)

712 #
713 # ISO9660 stage 1.5
714 #
715 $(ISO9660_STAGE1_5_EXEC): $(ISO9660_STAGE1_5_ASMOBJS) $(ISO9660_STAGE1_5_OBJS)
716         $(RM) $@
717         $(LINK) -o $@ $(ISO9660_STAGE1_5_ASMOBJS) $(ISO9660_STAGE1_5_OBJS) \
718             $(LIBS)

720 $(ISO9660_STAGE1_5_ASMOBJS): $$(@:iso9660_stage1_5_exec-%.o=%.S)
```



```
721         $(CCAS) $(CCASFLAGS) -c -o $@ $(@:iso9660_stage1_5_exec-%.o=%.S)

723 $(ISO9660_STAGE1_5_OBJS): $$(@:iso9660_stage1_5_exec-%.o=%.c)
724         $(CC) $(CFLAGS) -c -o $@ $(@:iso9660_stage1_5_exec-%.o=%.c)

726 #
727 # JFS stage 1.5
728 #
729 $(JFS_STAGE1_5_EXEC): $(JFS_STAGE1_5_ASMOBJS) $(JFS_STAGE1_5_OBJS)
730         $(RM) $@
731         $(LINK) -o $@ $(JFS_STAGE1_5_ASMOBJS) $(JFS_STAGE1_5_OBJS) $(LIBS)

733 $(JFS_STAGE1_5_ASMOBJS): $$(@:jfs_stage1_5_exec-%.o=%.S)
734         $(CCAS) $(CCASFLAGS) -c -o $@ $(@:jfs_stage1_5_exec-%.o=%.S)

736 $(JFS_STAGE1_5_OBJS): $$(@:jfs_stage1_5_exec-%.o=%.c)
737         $(CC) $(CFLAGS) -c -o $@ $(@:jfs_stage1_5_exec-%.o=%.c)

739 #
740 # Minix stage 1.5
741 #
742 $(MINIX_STAGE1_5_EXEC): $(MINIX_STAGE1_5_ASMOBJS) $(MINIX_STAGE1_5_OBJS)
743         $(RM) $@
744         $(LINK) -o $@ $(MINIX_STAGE1_5_ASMOBJS) $(MINIX_STAGE1_5_OBJS) $(LIBS)

746 $(MINIX_STAGE1_5_ASMOBJS): $$(@:minix_stage1_5_exec-%.o=%.S)
747         $(CCAS) $(CCASFLAGS) -c -o $@ $(@:minix_stage1_5_exec-%.o=%.S)

749 $(MINIX_STAGE1_5_OBJS): $$(@:minix_stage1_5_exec-%.o=%.c)
750         $(CC) $(CFLAGS) -c -o $@ $(@:minix_stage1_5_exec-%.o=%.c)

752 #
753 # nbgrub
754 #
755 $(NBGRUB_DATA): $(NBLOADER_DATA) $(DISKLESS_DATA)
756         $(RM) $@
757         cat $(NBLOADER_DATA) $(DISKLESS_DATA) > $@

759 #
760 # nbloader
761 #
762 $(NBLOADER_EXEC): $(NBLOADER_ASMOBJS) $(NBLOADER_OBJS)
763         $(RM) $@
764         $(LINK) -o $@ $(NBLOADER_ASMOBJS) $(NBLOADER_OBJS) $(LIBS)

766 $(NBLOADER_ASMOBJS): $$(@:nbloader_exec-%.o=%.S) diskless_size.h
767         $(CCAS) $(CCASFLAGS) -c -o $@ $(@:nbloader_exec-%.o=%.S)

769 $(NBLOADER_OBJS): $$(@:nbloader_exec-%.o=%.c)
770         $(CC) $(CFLAGS) -c -o $@ $(@:nbloader_exec-%.o=%.c)

772 #
773 # Pre-stage2
774 #
775 $(PRE_STAGE2_EXEC): $(PRE_STAGE2_ASMOBJS) $(PRE_STAGE2_OBJS)
776         $(RM) $@
777         $(LINK) -o $@ $(PRE_STAGE2_ASMOBJS) $(PRE_STAGE2_OBJS) $(LIBS)

779 $(PRE_STAGE2_ASMOBJS): $$(@:pre_stage2_exec-%.o=%.S)
780         $(CCAS) $(CCASFLAGS) -c -o $@ $(@:pre_stage2_exec-%.o=%.S)

782 $(PRE_STAGE2_OBJS): $$(@:pre_stage2_exec-%.o=%.c)
783         $(CC) $(CFLAGS) -c -o $@ $(@:pre_stage2_exec-%.o=%.c)

785 #
786 # pxegrub
```

```
787 #
788 $(PXEGRUB_DATA): $(PXELOADER_DATA) $(DISKLESS_DATA)
789         $(RM) $@
790         cat $(PXELOADER_DATA) $(DISKLESS_DATA) > $@

792 #
793 # pxeloader
794 #
795 $(PXELOADER_EXEC): $(PXELOADER_ASMOBJS) $(PXELOADER_OBJS)
796         $(RM) $@
797         $(LINK) -o $@ $(PXELOADER_ASMOBJS) $(PXELOADER_OBJS) $(LIBS)

799 $(PXELOADER_ASMOBJS): $$(@:pxeloader_exec-%.o=%.S) diskless_size.h
800         $(CCAS) $(CCASFLAGS) -c -o $@ $(@:pxeloader_exec-%.o=%.S)

802 $(PXELOADER_OBJS): $$(@:pxeloader_exec-%.o=%.c)
803         $(CC) $(CFLAGS) -c -o $@ $(@:pxeloader_exec-%.o=%.c)

805 #
806 # ReiserFS stage 1.5
807 #
808 $(REISERFS_STAGE1_5_EXEC): $(REISERFS_STAGE1_5_ASMOBJS) \
809         $(REISERFS_STAGE1_5_OBJS)
810         $(RM) $@
811         $(LINK) -o $@ $(REISERFS_STAGE1_5_ASMOBJS) $(REISERFS_STAGE1_5_OBJS) \
812             $(LIBS)

814 $(REISERFS_STAGE1_5_ASMOBJS): $$(@:reiserfs_stage1_5_exec-%.o=%.S)
815         $(CCAS) $(CCASFLAGS) -c -o $@ $(@:reiserfs_stage1_5_exec-%.o=%.S)

817 $(REISERFS_STAGE1_5_OBJS): $$(@:reiserfs_stage1_5_exec-%.o=%.c)
818         $(CC) $(CFLAGS) -c -o $@ $(@:reiserfs_stage1_5_exec-%.o=%.c)

820 #
821 # stage2
822 #
823 $(STAGE2_DATA): $(START_DATA) $(PRE_STAGE2_DATA)
824         $(RM) $@
825         cat $(START_DATA) $(PRE_STAGE2_DATA) > $@
826 #
827 # Eltorito stage2
828 #
829 $(STAGE2_ELTORITO_DATA): $(START_ELTORITO_DATA) $(PRE_STAGE2_DATA)
830         $(RM) $@
831         cat $(START_ELTORITO_DATA) $(PRE_STAGE2_DATA) > $@

833 #
834 # start
835 #
836 $(START_EXEC): $(START_ASMOBJS) $(START_OBJS)
837         $(RM) $@
838         $(LINK) -o $@ $(START_ASMOBJS) $(START_OBJS) $(LIBS)

840 $(START_ASMOBJS): $$(@:start_exec-%.o=%.S) stage2_size.h
841         $(CCAS) $(CCASFLAGS) -c -o $@ $(@:start_exec-%.o=%.S)

843 $(START_OBJS): $$(@:start_exec-%.o=%.c)
844         $(CC) $(CFLAGS) -c -o $@ $(@:start_exec-%.o=%.c)

846 #
847 # Eltorito start
848 #
849 $(START_ELTORITO_EXEC): $(START_ELTORITO_ASMOBJS) $(START_ELTORITO_OBJS)
850         $(RM) $@
851         $(LINK) -o $@ $(START_ELTORITO_ASMOBJS) $(START_ELTORITO_OBJS) $(LIBS)
```



```
853 $(START_ELTORITO_ASMOBJS): $$(@:start_eltorito_exec-%.o=%.S) stage2_size.h
854         $(CCAS) $(CCASFLAGS) -c -o $@ $(@:start_eltorito_exec-%.o=%.S)

856 $(START_ELTORITO_OBJS): $$(@:start_eltorito_exec-%.o=%.c)
857         $(CC) $(CFLAGS) -c -o $@ $(@:start_eltorito_exec-%.o=%.c)

859 #
860 # UFS2 stage 1.5
861 #
862 $(UFS2_STAGE1_5_EXEC): $(UFS2_STAGE1_5_ASMOBJS) $(UFS2_STAGE1_5_OBJS)
863         $(RM) $@
864         $(LINK) -o $@ $(UFS2_STAGE1_5_ASMOBJS) $(UFS2_STAGE1_5_OBJS) $(LIBS)

866 $(UFS2_STAGE1_5_ASMOBJS): $$(@:ufs2_stage1_5_exec-%.o=%.S)
867         $(CCAS) $(CCASFLAGS) -c -o $@ $(@:ufs2_stage1_5_exec-%.o=%.S)

869 $(UFS2_STAGE1_5_OBJS): $$(@:ufs2_stage1_5_exec-%.o=%.c)
870         $(CC) $(CFLAGS) -c -o $@ $(@:ufs2_stage1_5_exec-%.o=%.c)

872 #
873 # UFS stage 1.5
874 #
875 $(UFS_STAGE1_5_EXEC): $(UFS_STAGE1_5_ASMOBJS) $(UFS_STAGE1_5_OBJS)
876         $(RM) $@
877         $(LINK) -o $@ $(UFS_STAGE1_5_ASMOBJS) $(UFS_STAGE1_5_OBJS) $(LIBS)

879 $(UFS_STAGE1_5_ASMOBJS): $$(@:ufs_stage1_5_exec-%.o=%.S)
880         $(CCAS) $(CCASFLAGS) -c -o $@ $(@:ufs_stage1_5_exec-%.o=%.S)

882 $(UFS_STAGE1_5_OBJS): $$(@:ufs_stage1_5_exec-%.o=%.c)
883         $(CC) $(CFLAGS) -c -o $@ $(@:ufs_stage1_5_exec-%.o=%.c)

885 #
886 # ZFS stage 1.5
887 #
888 $(ZFS_STAGE1_5_EXEC): $(ZFS_STAGE1_5_ASMOBJS) $(ZFS_STAGE1_5_OBJS)
889         $(RM) $@
890         $(LINK) -o $@ $(ZFS_STAGE1_5_ASMOBJS) $(ZFS_STAGE1_5_OBJS) $(LIBS)

892 $(ZFS_STAGE1_5_ASMOBJS): $$(@:zfs_stage1_5_exec-%.o=%.S)
893         $(CCAS) $(CCASFLAGS) -c -o $@ $(@:zfs_stage1_5_exec-%.o=%.S)

895 $(ZFS_STAGE1_5_OBJS): $$(@:zfs_stage1_5_exec-%.o=%.c)
896         $(CC) $(CFLAGS) -c -o $@ $(@:zfs_stage1_5_exec-%.o=%.c)

898 #
899 # VSTa stage 1.5
900 #
901 $(VSTAFS_STAGE1_5_EXEC): $(VSTAFS_STAGE1_5_ASMOBJS) $(VSTAFS_STAGE1_5_OBJS)
902         $(RM) $@
903         $(LINK) -o $@ $(VSTAFS_STAGE1_5_ASMOBJS) $(VSTAFS_STAGE1_5_OBJS) $(LIBS)

905 $(VSTAFS_STAGE1_5_ASMOBJS): $$(@:vstafs_stage1_5_exec-%.o=%.S)
906         $(CCAS) $(CCASFLAGS) -c -o $@ $(@:vstafs_stage1_5_exec-%.o=%.S)

908 $(VSTAFS_STAGE1_5_OBJS): $$(@:vstafs_stage1_5_exec-%.o=%.c)
909         $(CC) $(CFLAGS) -c -o $@ $(@:vstafs_stage1_5_exec-%.o=%.c)

911 #
912 # SGI XFS stage 1.5
913 #
914 $(XFS_STAGE1_5_EXEC): $(XFS_STAGE1_5_ASMOBJS) $(XFS_STAGE1_5_OBJS)
915         $(RM) $@
916         $(LINK) -o $@ $(XFS_STAGE1_5_ASMOBJS) $(XFS_STAGE1_5_OBJS) $(LIBS)

918 $(XFS_STAGE1_5_ASMOBJS): $$(@:xfs_stage1_5_exec-%.o=%.S)
```

```
 919         $(CCAS) $(CCASFLAGS) -c -o $@ $(@:xfs_stage1_5_exec-%.o=%.S)

 921 $(XFS_STAGE1_5_OBJS): $$(@:xfs_stage1_5_exec-%.o=%.c)
 922         $(CC) $(CFLAGS) -c -o $@ $(@:xfs_stage1_5_exec-%.o=%.c)



 926 install: all $(INSTALL_TARGETS)

 928 $(ROOT_BOOT_GRUB)/%: $(ROOT_BOOT_GRUB) %
 929         $(INS.file)

 931 $(SRC_DIR)/%: $(SRC_DIR) %
 932         $(INS.file)

 934 $(SRC_ZFSINC_DIR)/%: $(SRC_ZFSINC_DIR) %
 935         $(INS.file)

 937 $(ROOT_BOOT_GRUB) $(SRC_DIR) $(SRC_ZFSINC_DIR):
 938         $(INS.dir)

 940 clean: _FORCE
 941         $(RM) *.o $(BUILT_SOURCES) $(LIBRARIES) $(PROGRAMS) $(NOINSTALL_DATA)

 943 clobber: clean
 944         $(RM) $(INSTALL_DATA)

 946 _FORCE:
```

```
**********************************************************
   43502 Tue Jan 15 15:38:22 2013
new/usr/src/grub/grub-0.97/stage2/fsys_zfs.c
%B
**********************************************************
```

```
/*
 *  GRUB  --  GRand Unified Bootloader
 *  Copyright (C) 1999,2000,2001,2002,2003,2004  Free Software Foundation, Inc.
 *
 *  This program is free software; you can redistribute it and/or modify
 *  it under the terms of the GNU General Public License as published by
 *  the Free Software Foundation; either version 2 of the License, or
 *  (at your option) any later version.
 *
 *  This program is distributed in the hope that it will be useful,
 *  but WITHOUT ANY WARRANTY; without even the implied warranty of
 *  MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
 *  GNU General Public License for more details.
 *
 *  You should have received a copy of the GNU General Public License
 *  along with this program; if not, write to the Free Software
 *  Foundation, Inc., 675 Mass Ave, Cambridge, MA 02139, USA.
 */

/*
 * Copyright 2010 Sun Microsystems, Inc.  All rights reserved.
 * Use is subject to license terms.
 */

/*
 * Copyright (c) 2012 by Delphix. All rights reserved.
 * Copyright (c) 2013 by Saso Kiselkov. All rights reserved.
 */

/*
 * The zfs plug-in routines for GRUB are:
 *
 * zfs_mount() - locates a valid uberblock of the root pool and reads
 *               in its MOS at the memory address MOS.
 *
 * zfs_open() - locates a plain file object by following the MOS
 *              and places its dnode at the memory address DNODE.
 *
 * zfs_read() - read in the data blocks pointed by the DNODE.
 *
 * ZFS_SCRATCH is used as a working area.
 *
 * (memory addr)   MOS      DNODE       ZFS_SCRATCH
 *                  |         |          |
 *          +-------V---------V----------V---------------+
 *   memory |       | dnode   | dnode    |  scratch      |
 *          |       | 512B    | 512B     |  area         |
 *          +-----------------------------------------+
 */

#ifdef  FSYS_ZFS

#include "shared.h"
#include "filesys.h"
#include "fsys_zfs.h"

/* cache for a file block of the currently zfs_open()-ed file */
static void *file_buf = NULL;
static uint64_t file_start = 0;
static uint64_t file_end = 0;
```



```
/* cache for a dnode block */
static dnode_phys_t *dnode_buf = NULL;
static dnode_phys_t *dnode_mdn = NULL;
static uint64_t dnode_start = 0;
static uint64_t dnode_end = 0;

static uint64_t pool_guid = 0;
static uberblock_t current_uberblock;
static char *stackbase;

decomp_entry_t decomp_table[ZIO_COMPRESS_FUNCTIONS] =
{
        {"inherit", 0},                 /* ZIO_COMPRESS_INHERIT */
        {"on", lzjb_decompress},        /* ZIO_COMPRESS_ON */
        {"off", 0},                     /* ZIO_COMPRESS_OFF */
        {"lzjb", lzjb_decompress},      /* ZIO_COMPRESS_LZJB */
        {"empty", 0},                   /* ZIO_COMPRESS_EMPTY */
        {"gzip-1", 0},                  /* ZIO_COMPRESS_GZIP_1 */
        {"gzip-2", 0},                  /* ZIO_COMPRESS_GZIP_2 */
        {"gzip-3", 0},                  /* ZIO_COMPRESS_GZIP_3 */
        {"gzip-4", 0},                  /* ZIO_COMPRESS_GZIP_4 */
        {"gzip-5", 0},                  /* ZIO_COMPRESS_GZIP_5 */
        {"gzip-6", 0},                  /* ZIO_COMPRESS_GZIP_6 */
        {"gzip-7", 0},                  /* ZIO_COMPRESS_GZIP_7 */
        {"gzip-8", 0},                  /* ZIO_COMPRESS_GZIP_8 */
        {"gzip-9", 0},                  /* ZIO_COMPRESS_GZIP_9 */
        {"zle", 0},                     /* ZIO_COMPRESS_ZLE */
        {"lz4", lz4_decompress}         /* ZIO_COMPRESS_LZ4 */
        {"empty", 0}                    /* ZIO_COMPRESS_EMPTY */
};
_____unchanged_portion_omitted_

/*
 * Read in a block of data, verify its checksum, decompress if needed,
 * and put the uncompressed data in buf.
 *
 * Return:
 *      0 - success
 *      errnum - failure
 */
static int
zio_read(blkptr_t *bp, void *buf, char *stack)
{
        int lsize, psize, comp;
        char *retbuf;

        comp = BP_GET_COMPRESS(bp);
        lsize = BP_GET_LSIZE(bp);
        psize = BP_GET_PSIZE(bp);

        if ((unsigned int)comp >= ZIO_COMPRESS_FUNCTIONS ||
            (comp != ZIO_COMPRESS_OFF &&
            decomp_table[comp].decomp_func == NULL)) {
                grub_printf("compression algorithm not supported\n");
                return (ERR_FSYS_CORRUPT);
        }

        if ((char *)buf < stack && ((char *)buf) + lsize > stack) {
                grub_printf("not enough memory allocated\n");
                return (ERR_WONT_FIT);
        }

        retbuf = buf;
        if (comp != ZIO_COMPRESS_OFF) {
                buf = stack;
                stack += psize;
```

```
 415         }

 417         if (zio_read_data(bp, buf, stack) != 0) {
 418                 grub_printf("zio_read_data failed\n");
 419                 return (ERR_FSYS_CORRUPT);
 420         }

 422         if (zio_checksum_verify(bp, buf, psize) != 0) {
 423                 grub_printf("checksum verification failed\n");
 424                 return (ERR_FSYS_CORRUPT);
 425         }

 427         if (comp != ZIO_COMPRESS_OFF) {
 428                 if (decomp_table[comp].decomp_func(buf, retbuf, psize,
 429                     lsize) != 0) {
 430                         grub_printf("zio_read decompression failed\n");
 431                         return (ERR_FSYS_CORRUPT);
 432                 }
 433         }
 415         if (comp != ZIO_COMPRESS_OFF)
 416                 decomp_table[comp].decomp_func(buf, retbuf, psize, lsize);

 435         return (0);
 436 }
______unchanged_portion_omitted_

 959 /*
 960  * List of pool features that the grub implementation of ZFS supports for
 961  * read. Note that features that are only required for write do not need
 962  * to be listed here since grub opens pools in read-only mode.
 963  */
 964 static const char *spa_feature_names[] = {
 965         "org.illumos:lz4_compress",
 966         NULL
 967 };
______unchanged_portion_omitted_
```

```
**********************************************************
    6419 Tue Jan 15 15:38:22 2013
new/usr/src/grub/grub-0.97/stage2/fsys_zfs.h
%B
**********************************************************
```

```
/*
 *  GRUB  --  GRand Unified Bootloader
 *  Copyright (C) 1999,2000,2001,2002,2003,2004  Free Software Foundation, Inc.
 *
 *  This program is free software; you can redistribute it and/or modify
 *  it under the terms of the GNU General Public License as published by
 *  the Free Software Foundation; either version 2 of the License, or
 *  (at your option) any later version.
 *
 *  This program is distributed in the hope that it will be useful,
 *  but WITHOUT ANY WARRANTY; without even the implied warranty of
 *  MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
 *  GNU General Public License for more details.
 *
 *  You should have received a copy of the GNU General Public License
 *  along with this program; if not, write to the Free Software
 *  Foundation, Inc., 675 Mass Ave, Cambridge, MA 02139, USA.
 */

/*
 * Copyright 2010 Sun Microsystems, Inc.  All rights reserved.
 * Use is subject to license terms.
 */

/*
 * Copyright (c) 2012 by Delphix. All rights reserved.
 * Copyright (c) 2013 by Saso Kiselkov. All rights reserved.
 */

#ifndef _FSYS_ZFS_H
#define _FSYS_ZFS_H

#ifdef  FSYS_ZFS

#ifndef FSIMAGE
typedef unsigned long long uint64_t;
typedef unsigned int uint32_t;
typedef unsigned short uint16_t;
typedef unsigned char uint8_t;
typedef unsigned char uchar_t;

#if defined(_LP64) || defined(_I32LPx)
typedef unsigned long size_t;
#else
typedef unsigned int size_t;
#endif
#else
#include "fsi_zfs.h"
#endif  /* !FSIMAGE */

#include <zfs-include/zfs.h>
#include <zfs-include/dmu.h>
#include <zfs-include/spa.h>
#include <zfs-include/zio.h>
#include <zfs-include/zio_checksum.h>
#include <zfs-include/vdev_impl.h>
#include <zfs-include/zap_impl.h>
#include <zfs-include/zap_leaf.h>
#include <zfs-include/uberblock_impl.h>
#include <zfs-include/dnode.h>
#include <zfs-include/dsl_dir.h>
```



```
#include <zfs-include/zfs_acl.h>
#include <zfs-include/zfs_znode.h>
#include <zfs-include/dsl_dataset.h>
#include <zfs-include/zil.h>
#include <zfs-include/dmu_objset.h>
#include <zfs-include/sa_impl.h>

/*
 * Global Memory addresses to store MOS and DNODE data
 */
#define MOS             ((dnode_phys_t *)\
        (RAW_ADDR((mbi.mem_upper << 10) + 0x100000) - ZFS_SCRATCH_SIZE))
#define DNODE           (MOS+1) /* move sizeof(dnode_phys_t) bytes */
#define ZFS_SCRATCH     ((char *)(DNODE+1))

/*
 * Verify dnode type.
 * Can only be used in functions returning non-0 for failure.
 */
#define VERIFY_DN_TYPE(dnp, type) \
        if (type && (dnp)->dn_type != type) { \
                return (ERR_FSYS_CORRUPT); \
        }

/*
 * Verify object set type.
 * Can only be used in functions returning 0 for failure.
 */
#define VERIFY_OS_TYPE(osp, type) \
        if (type && (osp)->os_type != type) { \
                errnum = ERR_FSYS_CORRUPT; \
                return (0); \
        }

#define ZPOOL_PROP_BOOTFS               "bootfs"

/* General macros */
#define BSWAP_8(x)      ((x) & 0xff)
#define BSWAP_16(x)     ((BSWAP_8(x) << 8) | BSWAP_8((x) >> 8))
#define BSWAP_32(x)     ((BSWAP_16(x) << 16) | BSWAP_16((x) >> 16))
#define BSWAP_64(x)     ((BSWAP_32(x) << 32) | BSWAP_32((x) >> 32))
#define P2ROUNDUP(x, align)     (-(-(x) & -(align)))

typedef struct uberblock uberblock_t;

/*
 * Macros to get fields in a bp or DVA.
 */
#define P2PHASE(x, align)               ((x) & ((align) - 1))
#define DVA_OFFSET_TO_PHYS_SECTOR(offset) \
        ((offset + VDEV_LABEL_START_SIZE) >> SPA_MINBLOCKSHIFT)

/*
 * return x rounded down to an align boundary
 * eg, P2ALIGN(1200, 1024) == 1024 (1*align)
 * eg, P2ALIGN(1024, 1024) == 1024 (1*align)
 * eg, P2ALIGN(0x1234, 0x100) == 0x1200 (0x12*align)
 * eg, P2ALIGN(0x5600, 0x100) == 0x5600 (0x56*align)
 */
#define P2ALIGN(x, align)               ((x) & -(align))

/*
 * For nvlist manipulation. (from nvpair.h)
 */
#define NV_ENCODE_NATIVE        0
#define NV_ENCODE_XDR           1
```

```
128 #define HOST_ENDIAN             1       /* for x86 machine */
129 typedef enum {
130         DATA_TYPE_UNKNOWN = 0,
131         DATA_TYPE_BOOLEAN,
132         DATA_TYPE_BYTE,
133         DATA_TYPE_INT16,
134         DATA_TYPE_UINT16,
135         DATA_TYPE_INT32,
136         DATA_TYPE_UINT32,
137         DATA_TYPE_INT64,
138         DATA_TYPE_UINT64,
139         DATA_TYPE_STRING,
140         DATA_TYPE_BYTE_ARRAY,
141         DATA_TYPE_INT16_ARRAY,
142         DATA_TYPE_UINT16_ARRAY,
143         DATA_TYPE_INT32_ARRAY,
144         DATA_TYPE_UINT32_ARRAY,
145         DATA_TYPE_INT64_ARRAY,
146         DATA_TYPE_UINT64_ARRAY,
147         DATA_TYPE_STRING_ARRAY,
148         DATA_TYPE_HRTIME,
149         DATA_TYPE_NVLIST,
150         DATA_TYPE_NVLIST_ARRAY,
151         DATA_TYPE_BOOLEAN_VALUE,
152         DATA_TYPE_INT8,
153         DATA_TYPE_UINT8,
154         DATA_TYPE_BOOLEAN_ARRAY,
155         DATA_TYPE_INT8_ARRAY,
156         DATA_TYPE_UINT8_ARRAY,
157         DATA_TYPE_DOUBLE
158 } data_type_t;
_____unchanged_portion_omitted_

174 /*
175  * FAT ZAP data structures
176  */
177 #define ZFS_CRC64_POLY 0xC96C5795D7870F42ULL /* ECMA-182, reflected form */
178 #define ZAP_HASH_IDX(hash, n)   (((n) == 0) ? 0 : ((hash) >> (64 - (n))))
179 #define CHAIN_END       0xffff  /* end of the chunk chain */

181 /*
182  * The amount of space within the chunk available for the array is:
183  * chunk size - space for type (1) - space for next pointer (2)
184  */
185 #define ZAP_LEAF_ARRAY_BYTES (ZAP_LEAF_CHUNKSIZE - 3)

187 #define ZAP_LEAF_HASH_SHIFT(bs) (bs - 5)
188 #define ZAP_LEAF_HASH_NUMENTRIES(bs) (1 << ZAP_LEAF_HASH_SHIFT(bs))
189 #define LEAF_HASH(bs, h) \
190         ((ZAP_LEAF_HASH_NUMENTRIES(bs)-1) & \
191         ((h) >> (64 - ZAP_LEAF_HASH_SHIFT(bs)-l->l_hdr.lh_prefix_len)))

193 /*
194  * The amount of space available for chunks is:
195  * block size shift - hash entry size (2) * number of hash
196  * entries - header space (2*chunksize)
197  */
198 #define ZAP_LEAF_NUMCHUNKS(bs) \
199         (((1<<bs) - 2*ZAP_LEAF_HASH_NUMENTRIES(bs)) / \
200         ZAP_LEAF_CHUNKSIZE - 2)

202 /*
203  * The chunks start immediately after the hash table.  The end of the
204  * hash table is at l_hash + HASH_NUMENTRIES, which we simply cast to a
205  * chunk_t.
206  */
```



```
207 #define ZAP_LEAF_CHUNK(l, bs, idx) \
208         ((zap_leaf_chunk_t *)(l->l_hash + ZAP_LEAF_HASH_NUMENTRIES(bs)))[idx]
209 #define ZAP_LEAF_ENTRY(l, bs, idx) (&ZAP_LEAF_CHUNK(l, bs, idx).l_entry)

211 extern void fletcher_2_native(const void *, uint64_t, zio_cksum_t *);
212 extern void fletcher_2_byteswap(const void *, uint64_t, zio_cksum_t *);
213 extern void fletcher_4_native(const void *, uint64_t, zio_cksum_t *);
214 extern void fletcher_4_byteswap(const void *, uint64_t, zio_cksum_t *);
215 extern void zio_checksum_SHA256(const void *, uint64_t, zio_cksum_t *);
216 extern int lzjb_decompress(void *, void *, size_t, size_t);
217 extern int lz4_decompress(void *, void *, size_t, size_t);

219 #endif  /* FSYS_ZFS */

221 #endif /* !_FSYS_ZFS_H */
```

```
**********************************************************
    2550 Tue Jan 15 15:38:22 2013
new/usr/src/grub/grub-0.97/stage2/zfs-include/zio.h
%B
**********************************************************
```

```
/*
 *  GRUB  --  GRand Unified Bootloader
 *  Copyright (C) 1999,2000,2001,2002,2003,2004  Free Software Foundation, Inc.
 *
 *  This program is free software; you can redistribute it and/or modify
 *  it under the terms of the GNU General Public License as published by
 *  the Free Software Foundation; either version 2 of the License, or
 *  (at your option) any later version.
 *
 *  This program is distributed in the hope that it will be useful,
 *  but WITHOUT ANY WARRANTY; without even the implied warranty of
 *  MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
 *  GNU General Public License for more details.
 *
 *  You should have received a copy of the GNU General Public License
 *  along with this program; if not, write to the Free Software
 *  Foundation, Inc., 675 Mass Ave, Cambridge, MA 02139, USA.
 */
/*
 * Copyright 2010 Sun Microsystems, Inc.  All rights reserved.
 * Use is subject to license terms.
 */
/*
 * Copyright 2013 by Saso Kiselkov. All rights reserved.
 */

#ifndef _ZIO_H
#define _ZIO_H

#define ZEC_MAGIC       0x210da7ab10c7a11ULL    /* zio data bloc tail */

typedef struct zio_eck {
        uint64_t        zec_magic;      /* for validation, endianness   */
        zio_cksum_t     zec_cksum;      /* 256-bit checksum             */
} zio_eck_t;
```

*unchanged_portion_omitted_*

```
enum zio_compress {
        ZIO_COMPRESS_INHERIT = 0,
        ZIO_COMPRESS_ON,
        ZIO_COMPRESS_OFF,
        ZIO_COMPRESS_LZJB,
        ZIO_COMPRESS_EMPTY,
        ZIO_COMPRESS_GZIP_1,
        ZIO_COMPRESS_GZIP_2,
        ZIO_COMPRESS_GZIP_3,
        ZIO_COMPRESS_GZIP_4,
        ZIO_COMPRESS_GZIP_5,
        ZIO_COMPRESS_GZIP_6,
        ZIO_COMPRESS_GZIP_7,
        ZIO_COMPRESS_GZIP_8,
        ZIO_COMPRESS_GZIP_9,
        ZIO_COMPRESS_ZLE,
        ZIO_COMPRESS_LZ4,
        ZIO_COMPRESS_FUNCTIONS
};
```

*unchanged_portion_omitted_*

```
**********************************************************
    8320 Tue Jan 15 15:38:22 2013
new/usr/src/grub/grub-0.97/stage2/zfs_lz4.c
%B
**********************************************************
```

```
/*
 * LZ4 - Fast LZ compression algorithm
 * Header File
 * Copyright (C) 2011-2013, Yann Collet.
 * BSD 2-Clause License (http://www.opensource.org/licenses/bsd-license.php)
 *
 * Redistribution and use in source and binary forms, with or without
 * modification, are permitted provided that the following conditions are
 * met:
 *
 *     * Redistributions of source code must retain the above copyright
 * notice, this list of conditions and the following disclaimer.
 *     * Redistributions in binary form must reproduce the above
 * copyright notice, this list of conditions and the following disclaimer
 * in the documentation and/or other materials provided with the
 * distribution.
 *
 * THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS
 * "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT
 * LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR
 * A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT
 * OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL,
 * SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT
 * LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE,
 * DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY
 * THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT
 * (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE
 * OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.
 *
 * You can contact the author at :
 * - LZ4 homepage : http://fastcompression.blogspot.com/p/lz4.html
 * - LZ4 source repository : http://code.google.com/p/lz4/
 */

#include "fsys_zfs.h"
#include <string.h>

static int LZ4_uncompress_unknownOutputSize(const char *source, char *dest,
                                            int isize, int maxOutputSize);

int
lz4_decompress(void *s_start, void *d_start, size_t s_len, size_t d_len)
{
        const uint8_t *src = s_start;
        uint32_t bufsiz = (src[0] << 24) | (src[1] << 16) | (src[2] << 8) |
            src[3];

        /* invalid compressed buffer size encoded at start */
        if (bufsiz + 4 > s_len)
                return (1);

        /*
         * Returns 0 on success (decompression function returned non-negative)
         * and non-zero on failure (decompression function returned negative).
         */
        return (LZ4_uncompress_unknownOutputSize(s_start + 4, d_start, bufsiz,
            d_len) < 0);
}

/*
 * CPU Feature Detection
```



```
 */

/* 32 or 64 bits ? */
#if (defined(__x86_64__) || defined(__x86_64) || defined(__amd64__) || \
        defined(__amd64) || defined(__ppc64__) || defined(_WIN64) || \
        defined(__LP64__) || defined(_LP64))
#define LZ4_ARCH64      1
#else
#define LZ4_ARCH64      0
#endif

/*
 * Little Endian or Big Endian?
 * Note: overwrite the below #define if you know your architecture endianess.
 */
#if (defined(__BIG_ENDIAN__) || defined(__BIG_ENDIAN) || \
        defined(_BIG_ENDIAN) || defined(_ARCH_PPC) || defined(__PPC__) || \
        defined(__PPC) || defined(PPC) || defined(__powerpc__) || \
        defined(__powerpc) || defined(powerpc) || \
        ((defined(__BYTE_ORDER__)&&(__BYTE_ORDER__ == __ORDER_BIG_ENDIAN__))))
#define LZ4_BIG_ENDIAN  1
#else
        /*
         * Little Endian assumed. PDP Endian and other very rare endian format
         * are unsupported.
         */
#endif

/*
 * Compiler Options
 */
#if __STDC_VERSION__ >= 199901L /* C99 */
/* "restrict" is a known keyword */
#else
/* Disable restrict */
#define restrict
#endif

#define GCC_VERSION (__GNUC__ * 100 + __GNUC_MINOR__)

#define lz4_bswap16(x) ((unsigned short int) ((((x) >> 8) & 0xffu) \
        | (((x) & 0xffu) << 8)))

#if (GCC_VERSION >= 302) || (__INTEL_COMPILER >= 800) || defined(__clang__)
#define expect(expr, value)    (__builtin_expect((expr), (value)))
#else
#define expect(expr, value)    (expr)
#endif

#define likely(expr)    expect((expr) != 0, 1)
#define unlikely(expr)  expect((expr) != 0, 0)

/* Basic types */
#define BYTE    uint8_t
#define U16     uint16_t
#define U32     uint32_t
#define S32     int32_t
#define U64     uint64_t

typedef struct _U16_S {
        U16 v;
} U16_S;
typedef struct _U32_S {
        U32 v;
} U32_S;
typedef struct _U64_S {
```

```
128         U64 v;
129 } U64_S;

131 #define A64(x)  (((U64_S *)(x))->v)
132 #define A32(x)  (((U32_S *)(x))->v)
133 #define A16(x)  (((U16_S *)(x))->v)

135 /*
136  * Constants
137  */
138 #define MINMATCH 4

140 #define COPYLENGTH 8
141 #define LASTLITERALS 5

143 #define ML_BITS 4
144 #define ML_MASK ((1U<<ML_BITS)-1)
145 #define RUN_BITS (8-ML_BITS)
146 #define RUN_MASK ((1U<<RUN_BITS)-1)

148 /*
149  * Architecture-specific macros
150  */
151 #if LZ4_ARCH64
152 #define STEPSIZE 8
153 #define UARCH U64
154 #define AARCH A64
155 #define LZ4_COPYSTEP(s, d)      A64(d) = A64(s); d += 8; s += 8;
156 #define LZ4_COPYPACKET(s, d)    LZ4_COPYSTEP(s, d)
157 #define LZ4_SECURECOPY(s, d, e) if (d < e) LZ4_WILDCOPY(s, d, e)
158 #define HTYPE U32
159 #define INITBASE(base)          const BYTE* const base = ip
160 #else
161 #define STEPSIZE 4
162 #define UARCH U32
163 #define AARCH A32
164 #define LZ4_COPYSTEP(s, d)      A32(d) = A32(s); d += 4; s += 4;
165 #define LZ4_COPYPACKET(s, d)    LZ4_COPYSTEP(s, d); LZ4_COPYSTEP(s, d);
166 #define LZ4_SECURECOPY          LZ4_WILDCOPY
167 #define HTYPE const BYTE*
168 #define INITBASE(base)          const int base = 0
169 #endif

171 #if (defined(LZ4_BIG_ENDIAN) && !defined(BIG_ENDIAN_NATIVE_BUT_INCOMPATIBLE))
172 #define LZ4_READ_LITTLEENDIAN_16(d, s, p) \
173         { U16 v = A16(p); v = lz4_bswap16(v); d = (s) - v; }
174 #define LZ4_WRITE_LITTLEENDIAN_16(p, i) \
175         { U16 v = (U16)(i); v = lz4_bswap16(v); A16(p) = v; p += 2; }
176 #else
177 #define LZ4_READ_LITTLEENDIAN_16(d, s, p) { d = (s) - A16(p); }
178 #define LZ4_WRITE_LITTLEENDIAN_16(p, v)  { A16(p) = v; p += 2; }
179 #endif

181 /* Macros */
182 #define LZ4_WILDCOPY(s, d, e) do { LZ4_COPYPACKET(s, d) } while (d < e);

184 /* Decompression functions */

186 static int
187 LZ4_uncompress_unknownOutputSize(const char *source,
188     char *dest, int isize, int maxOutputSize)
189 {
190         /* Local Variables */
191         const BYTE *restrict ip = (const BYTE *) source;
192         const BYTE *const iend = ip + isize;
193         const BYTE *restrict ref;
```



```
195         BYTE *restrict op = (BYTE *) dest;
196         BYTE *const oend = op + maxOutputSize;
197         BYTE *cpy;

199         size_t dec[] = { 0, 3, 2, 3, 0, 0, 0, 0 };

201         /* Main Loop */
202         while (ip < iend) {
203                 BYTE token;
204                 int length;

206                 /* get runlength */
207                 token = *ip++;
208                 if ((length = (token >> ML_BITS)) == RUN_MASK) {
209                         int s = 255;
210                         while ((ip < iend) && (s == 255)) {
211                                 s = *ip++;
212                                 length += s;
213                         }
214                 }
215                 /* copy literals */
216                 cpy = op + length;
217                 if ((cpy > oend - COPYLENGTH) ||
218                     (ip + length > iend - COPYLENGTH)) {
219                         if (cpy > oend)
220                                 /*
221                                  * Error: request to write beyond destination
222                                  * buffer.
223                                  */
224                                 goto _output_error;
225                         if (ip + length > iend)
226                                 /*
227                                  * Error : request to read beyond source
228                                  * buffer.
229                                  */
230                                 goto _output_error;
231                         memcpy(op, ip, length);
232                         op += length;
233                         ip += length;
234                         if (ip < iend)
235                                 /* Error : LZ4 format violation */
236                                 goto _output_error;
237                         /* Necessarily EOF, due to parsing restrictions. */
238                         break;
239                 }
240                 LZ4_WILDCOPY(ip, op, cpy);
241                 ip -= (op - cpy);
242                 op = cpy;

244                 /* get offset */
245                 LZ4_READ_LITTLEENDIAN_16(ref, cpy, ip);
246                 ip += 2;
247                 if (ref < (BYTE * const) dest)
248                         /*
249                          * Error: offset creates reference outside of
250                          * destination buffer.
251                          */
252                         goto _output_error;

254                 /* get matchlength */
255                 if ((length = (token & ML_MASK)) == ML_MASK) {
256                         while (ip < iend) {
257                                 int s = *ip++;
258                                 length += s;
259                                 if (s == 255)
```

```
260                                         continue;
261                                 break;
262                         }
263                 }
264                 /* copy repeated sequence */
265                 if unlikely(op - ref < STEPSIZE) {
266 #if LZ4_ARCH64
267                         size_t dec2table[] = { 0, 0, 0, -1, 0, 1, 2, 3 };
268                         size_t dec2 = dec2table[op - ref];
269 #else
270                         const int dec2 = 0;
271 #endif
272                         *op++ = *ref++;
273                         *op++ = *ref++;
274                         *op++ = *ref++;
275                         *op++ = *ref++;
276                         ref -= dec[op - ref];
277                         A32(op) = A32(ref);
278                         op += STEPSIZE - 4;
279                         ref -= dec2;
280                 } else {
281                         LZ4_COPYSTEP(ref, op);
282                 }
283                 cpy = op + length - (STEPSIZE - 4);
284                 if (cpy > oend - COPYLENGTH) {
285                         if (cpy > oend)
286                                 /*
287                                  * Error: request to write outside of
288                                  * destination buffer.
289                                  */
290                                 goto _output_error;
291                         LZ4_SECURECOPY(ref, op, (oend - COPYLENGTH));
292                         while (op < cpy)
293                                 *op++ = *ref++;
294                         op = cpy;
295                         if (op == oend)
296                                 /*
297                                  * Check EOF (should never happen, since last
298                                  * 5 bytes are supposed to be literals).
299                                  */
300                                 break;
301                         continue;
302                 }
303                 LZ4_SECURECOPY(ref, op, cpy);
304                 op = cpy;       /* correction */
305         }

307         /* end of decoding */
308         return (int)(((char *)op) - dest);

310         /* write overflow error detected */
311         _output_error:
312         return (int)(-(((char *)ip) - source));
313 }
```

```
**********************************************************
  109167 Tue Jan 15 15:38:22 2013
new/usr/src/man/man1m/zfs.1m
%B
**********************************************************
'\" t
.\"
.\" CDDL HEADER START
.\"
.\" The contents of this file are subject to the terms of the
.\" Common Development and Distribution License (the "License").
.\" You may not use this file except in compliance with the License.
.\"
.\" You can obtain a copy of the license at usr/src/OPENSOLARIS.LICENSE
.\" or http://www.opensolaris.org/os/licensing.
.\" See the License for the specific language governing permissions
.\" and limitations under the License.
.\"
.\" When distributing Covered Code, include this CDDL HEADER in each
.\" file and include the License file at usr/src/OPENSOLARIS.LICENSE.
.\" If applicable, add the following below this CDDL HEADER, with the
.\" fields enclosed by brackets "[]" replaced with your own identifying
.\" information: Portions Copyright [yyyy] [name of copyright owner]
.\"
.\" CDDL HEADER END
.\"
.\"
.\" Copyright (c) 2009 Sun Microsystems, Inc. All Rights Reserved.
.\" Copyright 2011 Joshua M. Clulow <josh@sysmgr.org>
.\" Copyright (c) 2012 by Delphix. All rights reserved.
.\" Copyright (c) 2012, Joyent, Inc. All rights reserved.
.\" Copyright 2012 Nexenta Systems, Inc. All Rights Reserved.
.\" Copyright (c) 2013 by Saso Kiselkov. All rights reserved.
.\"
.TH ZFS 1M "Sep 16, 2012"
.SH NAME
zfs \- configures ZFS file systems
.SH SYNOPSIS
.LP
.nf
\fBzfs\fR [\fB-?\fR]
.fi

.LP
.nf
\fBzfs\fR \fBcreate\fR [\fB-p\fR] [\fB-o\fR \fIproperty\fR=\fIvalue\fR] ... \fIf
.fi

.LP
.nf
\fBzfs\fR \fBcreate\fR [\fB-ps\fR] [\fB-b\fR \fIblocksize\fR] [\fB-o\fR \fIprope
.fi

.LP
.nf
\fBzfs\fR \fBdestroy\fR [\fB-fnpRrv\fR] \fIfilesystem\fR|\fIvolume\fR
.fi

.LP
.nf
\fBzfs\fR \fBdestroy\fR [\fB-dnpRrv\fR] \fIfilesystem\fR|\fIvolume\fR@\fIsnap\fR
.fi

.LP
.nf
\fBzfs\fR \fBsnapshot\fR [\fB-r\fR] [\fB-o\fR \fIproperty\fR=\fIvalue\fR]...
```



```
      \fIfilesystem@snapname\fR|\fIvolume@snapname\fR...
.fi

.LP
.nf
\fBzfs\fR \fBrollback\fR [\fB-rRf\fR] \fIsnapshot\fR
.fi

.LP
.nf
\fBzfs\fR \fBclone\fR [\fB-p\fR] [\fB-o\fR \fIproperty\fR=\fIvalue\fR] ... \fIsn
.fi

.LP
.nf
\fBzfs\fR \fBpromote\fR \fIclone-filesystem\fR
.fi

.LP
.nf
\fBzfs\fR \fBrename\fR [\fB-f\fR] \fIfilesystem\fR|\fIvolume\fR|\fIsnapshot\fR
     \fIfilesystem\fR|\fIvolume\fR|\fIsnapshot\fR
.fi

.LP
.nf
\fBzfs\fR \fBrename\fR [\fB-fp\fR] \fIfilesystem\fR|\fIvolume\fR \fIfilesystem\f
.fi

.LP
.nf
\fBzfs\fR \fBrename\fR \fB-r\fR \fIsnapshot\fR \fIsnapshot\fR
.fi

.LP
.nf
\fBzfs\fR \fBlist\fR [\fB-r\fR|\fB-d\fR \fIdepth\fR][\fB-H\fR][\fB-o\fR \fIprope
     [\fB-s\fR \fIproperty\fR] ... [\fB-S\fR \fIproperty\fR] ... [\fIfilesystem\
.fi

.LP
.nf
\fBzfs\fR \fBset\fR \fIproperty\fR=\fIvalue\fR \fIfilesystem\fR|\fIvolume\fR|\fI
.fi

.LP
.nf
\fBzfs\fR \fBget\fR [\fB-r\fR|\fB-d\fR \fIdepth\fR][\fB-Hp\fR][\fB-o\fR \fIfield
    [\fB-s\fR \fIsource\fR[,...]] "\fIall\fR" | \fIproperty\fR[,...] \fIfilesyst
.fi

.LP
.nf
\fBzfs\fR \fBinherit\fR [\fB-r\fR] \fIproperty\fR \fIfilesystem\fR|\fIvolume|sna
.fi

.LP
.nf
\fBzfs\fR \fBupgrade\fR [\fB-v\fR]
.fi

.LP
.nf
\fBzfs\fR \fBupgrade\fR [\fB-r\fR] [\fB-V\fR \fIversion\fR] \fB-a\fR | \fIfilesy
.fi
```

```
.LP
.nf
\fBzfs\fR \fBuserspace\fR [\fB-Hinp\fR] [\fB-o\fR \fIfield\fR[,...]] [\fB-s\fR \
    [\fB-S\fR \fIfield\fR] ... [\fB-t\fR \fItype\fR[,...]] \fIfilesystem\fR|\fIs
.fi

.LP
.nf
\fBzfs\fR \fBgroupspace\fR [\fB-Hinp\fR] [\fB-o\fR \fIfield\fR[,...]] [\fB-s\fR
    [\fB-S\fR \fIfield\fR] ... [\fB-t\fR \fItype\fR[,...]] \fIfilesystem\fR|\fIs
.fi

.LP
.nf
\fBzfs\fR \fBmount\fR
.fi

.LP
.nf
\fBzfs\fR \fBmount\fR [\fB-vO\fR] [\fB-o \fIoptions\fR\fR] \fB-a\fR | \fIfilesys
.fi

.LP
.nf
\fBzfs\fR \fBunmount\fR [\fB-f\fR] \fB-a\fR | \fIfilesystem\fR|\fImountpoint\fR
.fi

.LP
.nf
\fBzfs\fR \fBshare\fR \fB-a\fR | \fIfilesystem\fR
.fi

.LP
.nf
\fBzfs\fR \fBunshare\fR \fB-a\fR \fIfilesystem\fR|\fImountpoint\fR
.fi

.LP
.nf
\fBzfs\fR \fBsend\fR [\fB-DnPpRrv\fR] [\fB-\fR[\fBiI\fR] \fIsnapshot\fR] \fIsnap
.fi

.LP
.nf
\fBzfs\fR \fBreceive\fR [\fB-vnFu\fR] \fIfilesystem\fR|\fIvolume\fR|\fIsnapshot\
.fi

.LP
.nf
\fBzfs\fR \fBreceive\fR [\fB-vnFu\fR] [\fB-d\fR|\fB-e\fR] \fIfilesystem\fR
.fi

.LP
.nf
\fBzfs\fR \fBallow\fR \fIfilesystem\fR|\fIvolume\fR
.fi

.LP
.nf
\fBzfs\fR \fBallow\fR [\fB-ldug\fR] "\fIeveryone\fR"|\fIuser\fR|\fIgroup\fR[,...
     \fIfilesystem\fR|\fIvolume\fR
.fi

.LP
.nf
\fBzfs\fR \fBallow\fR [\fB-ld\fR] \fB-e\fR \fIperm\fR|@\fIsetname\fR[,...] \fIfi
```



```
.fi

.LP
.nf
\fBzfs\fR \fBallow\fR \fB-c\fR \fIperm\fR|@\fIsetname\fR[,...] \fIfilesystem\fR|
.fi

.LP
.nf
\fBzfs\fR \fBallow\fR \fB-s\fR @\fIsetname\fR \fIperm\fR|@\fIsetname\fR[,...] \f
.fi

.LP
.nf
\fBzfs\fR \fBunallow\fR [\fB-rldug\fR] "\fIeveryone\fR"|\fIuser\fR|\fIgroup\fR[,
     \fIfilesystem\fR|\fIvolume\fR
.fi

.LP
.nf
\fBzfs\fR \fBunallow\fR [\fB-rld\fR] \fB-e\fR [\fIperm\fR|@\fIsetname\fR[,... ]]
.fi

.LP
.nf
\fBzfs\fR \fBunallow\fR [\fB-r\fR] \fB-c\fR [\fIperm\fR|@\fIsetname\fR[ ... ]] \
.fi

.LP
.nf
\fBzfs\fR \fBunallow\fR [\fB-r\fR] \fB-s\fR @\fIsetname\fR [\fIperm\fR|@\fIsetna
.fi

.LP
.nf
\fBzfs\fR \fBhold\fR [\fB-r\fR] \fItag\fR \fIsnapshot\fR...
.fi

.LP
.nf
\fBzfs\fR \fBholds\fR [\fB-r\fR] \fIsnapshot\fR...
.fi

.LP
.nf
\fBzfs\fR \fBrelease\fR [\fB-r\fR] \fItag\fR \fIsnapshot\fR...
.fi

.LP
.nf
\fBzfs\fR \fBdiff\fR [\fB-FHt\fR] \fIsnapshot\fR \fIsnapshot|filesystem\fR

.SH DESCRIPTION
.sp
.LP
The \fBzfs\fR command configures \fBZFS\fR datasets within a \fBZFS\fR storage
pool, as described in \fBzpool\fR(1M). A dataset is identified by a unique path
within the \fBZFS\fR namespace. For example:
.sp
.in +2
.nf
pool/{filesystem,volume,snapshot}
.fi
.in -2
.sp
```

```
.sp
.LP
where the maximum length of a dataset name is \fBMAXNAMELEN\fR (256 bytes).
.sp
.LP
A dataset can be one of the following:
.sp
.ne 2
.na
\fB\fIfile system\fR\fR
.ad
.sp .6
.RS 4n
A \fBZFS\fR dataset of type \fBfilesystem\fR can be mounted within the standard
system namespace and behaves like other file systems. While \fBZFS\fR file
systems are designed to be \fBPOSIX\fR compliant, known issues exist that
prevent compliance in some cases. Applications that depend on standards
conformance might fail due to nonstandard behavior when checking file system
free space.
.RE

.sp
.ne 2
.na
\fB\fIvolume\fR\fR
.ad
.sp .6
.RS 4n
A logical volume exported as a raw or block device. This type of dataset should
only be used under special circumstances. File systems are typically used in
most environments.
.RE

.sp
.ne 2
.na
\fB\fIsnapshot\fR\fR
.ad
.sp .6
.RS 4n
A read-only version of a file system or volume at a given point in time. It is
specified as \fIfilesystem@name\fR or \fIvolume@name\fR.
.RE

.SS "ZFS File System Hierarchy"
.sp
.LP
A \fBZFS\fR storage pool is a logical collection of devices that provide space
for datasets. A storage pool is also the root of the \fBZFS\fR file system
hierarchy.
.sp
.LP
The root of the pool can be accessed as a file system, such as mounting and
unmounting, taking snapshots, and setting properties. The physical storage
characteristics, however, are managed by the \fBzpool\fR(1M) command.
.sp
.LP
See \fBzpool\fR(1M) for more information on creating and administering pools.
.SS "Snapshots"
.sp
.LP
A snapshot is a read-only copy of a file system or volume. Snapshots can be
created extremely quickly, and initially consume no additional space within the
pool. As data within the active dataset changes, the snapshot consumes more
data than would otherwise be shared with the active dataset.
.sp
```



```
.LP
Snapshots can have arbitrary names. Snapshots of volumes can be cloned or
rolled back, but cannot be accessed independently.
.sp
.LP
File system snapshots can be accessed under the \fB\&.zfs/snapshot\fR directory
in the root of the file system. Snapshots are automatically mounted on demand
and may be unmounted at regular intervals. The visibility of the \fB\&.zfs\fR
directory can be controlled by the \fBsnapdir\fR property.
.SS "Clones"
.sp
.LP
A clone is a writable volume or file system whose initial contents are the same
as another dataset. As with snapshots, creating a clone is nearly
instantaneous, and initially consumes no additional space.
.sp
.LP
Clones can only be created from a snapshot. When a snapshot is cloned, it
creates an implicit dependency between the parent and child. Even though the
clone is created somewhere else in the dataset hierarchy, the original snapshot
cannot be destroyed as long as a clone exists. The \fBorigin\fR property
exposes this dependency, and the \fBdestroy\fR command lists any such
dependencies, if they exist.
.sp
.LP
The clone parent-child dependency relationship can be reversed by using the
\fBpromote\fR subcommand. This causes the "origin" file system to become a
clone of the specified file system, which makes it possible to destroy the file
system that the clone was created from.
.SS "Mount Points"
.sp
.LP
Creating a \fBZFS\fR file system is a simple operation, so the number of file
systems per system is likely to be numerous. To cope with this, \fBZFS\fR
automatically manages mounting and unmounting file systems without the need to
edit the \fB/etc/vfstab\fR file. All automatically managed file systems are
mounted by \fBZFS\fR at boot time.
.sp
.LP
By default, file systems are mounted under \fB/\fIpath\fR\fR, where \fIpath\fR
is the name of the file system in the \fBZFS\fR namespace. Directories are
created and destroyed as needed.
.sp
.LP
A file system can also have a mount point set in the \fBmountpoint\fR property.
This directory is created as needed, and \fBZFS\fR automatically mounts the
file system when the \fBzfs mount -a\fR command is invoked (without editing
\fB/etc/vfstab\fR). The \fBmountpoint\fR property can be inherited, so if
\fBpool/home\fR has a mount point of \fB/export/stuff\fR, then
\fBpool/home/user\fR automatically inherits a mount point of
\fB/export/stuff/user\fR.
.sp
.LP
A file system \fBmountpoint\fR property of \fBnone\fR prevents the file system
from being mounted.
.sp
.LP
If needed, \fBZFS\fR file systems can also be managed with traditional tools
(\fBmount\fR, \fBumount\fR, \fB/etc/vfstab\fR). If a file system's mount point
is set to \fBlegacy\fR, \fBZFS\fR makes no attempt to manage the file system,
and the administrator is responsible for mounting and unmounting the file
system.
.SS "Zones"
.sp
.LP
A \fBZFS\fR file system can be added to a non-global zone by using the
```

\fBzonecfg\fR \fBadd fs\fR subcommand. A \fBZFS\fR file system that is added to a non-global zone must have its \fBmountpoint\fR property set to \fBlegacy\fR.

.LP

The physical properties of an added file system are controlled by the global administrator. However, the zone administrator can create, modify, or destroy files within the added file system, depending on how the file system is mounted.

.LP

A dataset can also be delegated to a non-global zone by using the \fBzonecfg\fR \fBadd dataset\fR subcommand. You cannot delegate a dataset to one zone and the children of the same dataset to another zone. The zone administrator can change properties of the dataset or any of its children. However, the \fBquota\fR property is controlled by the global administrator.

.LP

A \fBZFS\fR volume can be added as a device to a non-global zone by using the \fBzonecfg\fR \fBadd device\fR subcommand. However, its physical properties can be modified only by the global administrator.

.LP

For more information about \fBzonecfg\fR syntax, see \fBzonecfg\fR(1M).

.LP

After a dataset is delegated to a non-global zone, the \fBzoned\fR property is automatically set. A zoned file system cannot be mounted in the global zone, since the zone administrator might have to set the mount point to an unacceptable value.

.LP

The global administrator can forcibly clear the \fBzoned\fR property, though this should be done with extreme care. The global administrator should verify that all the mount points are acceptable before clearing the property.

.SS "Native Properties"

.LP

Properties are divided into two types, native properties and user-defined (or "user") properties. Native properties either export internal statistics or control \fBZFS\fR behavior. In addition, native properties are either editable or read-only. User properties have no effect on \fBZFS\fR behavior, but you can use them to annotate datasets in a way that is meaningful in your environment. For more information about user properties, see the "User Properties" section, below.

.LP

Every dataset has a set of properties that export statistics about the dataset as well as control various behaviors. Properties are inherited from the parent unless overridden by the child. Some properties apply only to certain types of datasets (file systems, volumes, or snapshots).

.LP

The values of numeric properties can be specified using human-readable suffixes (for example, \fBk\fR, \fBKB\fR, \fBM\fR, \fBGb\fR, and so forth, up to \fBZ\fR for zettabyte). The following are all valid (and equal) specifications:

.in +2

.nf

1536M, 1.5g, 1.50GB

.fi

.in -2

.sp

.LP

The values of non-numeric properties are case sensitive and must be lowercase,

except for \fBmountpoint\fR, \fBsharenfs\fR, and \fBsharesmb\fR.

.LP

The following native properties consist of read-only statistics about the dataset. These properties can be neither set, nor inherited. Native properties apply to all dataset types unless otherwise noted.

.ne 2

.na

\fB\fBavailable\fR\fR

.ad

.RS 4n

The amount of space available to the dataset and all its children, assuming that there is no other activity in the pool. Because space is shared within a pool, availability can be limited by any number of factors, including physical pool size, quotas, reservations, or other datasets within the pool.

This property can also be referred to by its shortened column name, \fBavail\fR.

.RE

.ne 2

.na

\fB\fBcompressratio\fR\fR

.ad

.RS 4n

For non-snapshots, the compression ratio achieved for the \fBused\fR space of this dataset, expressed as a multiplier. The \fBused\fR property includes descendant datasets, and, for clones, does not include the space shared with the origin snapshot. For snapshots, the \fBcompressratio\fR is the same as the \fBrefcompressratio\fR property. Compression can be turned on by running: \fBzfs set compression=on \fIdataset\fR\fR. The default value is \fBoff\fR.

.RE

.ne 2

.na

\fB\fBcreation\fR\fR

.ad

.RS 4n

The time this dataset was created.

.RE

.ne 2

.na

\fB\fBclones\fR\fR

.ad

.RS 4n

For snapshots, this property is a comma-separated list of filesystems or volumes which are clones of this snapshot. The clones' \fBorigin\fR property is this snapshot. If the \fBclones\fR property is not empty, then this snapshot can not be destroyed (even with the \fB-r\fR or \fB-f\fR options).

.RE

.ne 2

.na

\fB\fBdefer_destroy\fR\fR

.ad

```
524 .sp .6
525 .RS 4n
526 This property is \fBon\fR if the snapshot has been marked for deferred destroy
527 by using the \fBzfs destroy\fR \fB-d\fR command. Otherwise, the property is
528 \fBoff\fR.
529 .RE

531 .sp
532 .ne 2
533 .na
534 \fB\fBmounted\fR\fR
535 .ad
536 .sp .6
537 .RS 4n
538 For file systems, indicates whether the file system is currently mounted. This
539 property can be either \fByes\fR or \fBno\fR.
540 .RE

542 .sp
543 .ne 2
544 .na
545 \fB\fBorigin\fR\fR
546 .ad
547 .sp .6
548 .RS 4n
549 For cloned file systems or volumes, the snapshot from which the clone was
550 created.  See also the \fBclones\fR property.
551 .RE

553 .sp
554 .ne 2
555 .na
556 \fB\fBreferenced\fR\fR
557 .ad
558 .sp .6
559 .RS 4n
560 The amount of data that is accessible by this dataset, which may or may not be
561 shared with other datasets in the pool. When a snapshot or clone is created, it
562 initially references the same amount of space as the file system or snapshot it
563 was created from, since its contents are identical.
564 .sp
565 This property can also be referred to by its shortened column name,
566 \fBrefer\fR.
567 .RE

569 .sp
570 .ne 2
571 .na
572 \fB\fBrefcompressratio\fR\fR
573 .ad
574 .sp .6
575 .RS 4n
576 The compression ratio achieved for the \fBreferenced\fR space of this
577 dataset, expressed as a multiplier.  See also the \fBcompressratio\fR
578 property.
579 .RE

581 .sp
582 .ne 2
583 .na
584 \fB\fBtype\fR\fR
585 .ad
586 .sp .6
587 .RS 4n
588 The type of dataset: \fBfilesystem\fR, \fBvolume\fR, or \fBsnapshot\fR.
589 .RE
```



```
591 .sp
592 .ne 2
593 .na
594 \fB\fBused\fR\fR
595 .ad
596 .sp .6
597 .RS 4n
598 The amount of space consumed by this dataset and all its descendents. This is
599 the value that is checked against this dataset's quota and reservation. The
600 space used does not include this dataset's reservation, but does take into
601 account the reservations of any descendent datasets. The amount of space that a
602 dataset consumes from its parent, as well as the amount of space that are freed
603 if this dataset is recursively destroyed, is the greater of its space used and
604 its reservation.
605 .sp
606 When snapshots (see the "Snapshots" section) are created, their space is
607 initially shared between the snapshot and the file system, and possibly with
608 previous snapshots. As the file system changes, space that was previously
609 shared becomes unique to the snapshot, and counted in the snapshot's space
610 used. Additionally, deleting snapshots can increase the amount of space unique
611 to (and used by) other snapshots.
612 .sp
613 The amount of space used, available, or referenced does not take into account
614 pending changes. Pending changes are generally accounted for within a few
615 seconds. Committing a change to a disk using \fBfsync\fR(3c) or \fBO_SYNC\fR
616 does not necessarily guarantee that the space usage information is updated
617 immediately.
618 .RE

620 .sp
621 .ne 2
622 .na
623 \fB\fBusedby*\fR\fR
624 .ad
625 .sp .6
626 .RS 4n
627 The \fBusedby*\fR properties decompose the \fBused\fR properties into the
628 various reasons that space is used. Specifically, \fBused\fR =
629 \fBusedbychildren\fR + \fBusedbydataset\fR + \fBusedbyrefreservation\fR +,
630 \fBusedbysnapshots\fR. These properties are only available for datasets created
631 on \fBzpool\fR "version 13" pools.
632 .RE

634 .sp
635 .ne 2
636 .na
637 \fB\fBusedbychildren\fR\fR
638 .ad
639 .sp .6
640 .RS 4n
641 The amount of space used by children of this dataset, which would be freed if
642 all the dataset's children were destroyed.
643 .RE

645 .sp
646 .ne 2
647 .na
648 \fB\fBusedbydataset\fR\fR
649 .ad
650 .sp .6
651 .RS 4n
652 The amount of space used by this dataset itself, which would be freed if the
653 dataset were destroyed (after first removing any \fBrefreservation\fR and
654 destroying any necessary snapshots or descendents).
655 .RE
```

```
.sp
.ne 2
.na
\fB\fBusedbyrefreservation\fR\fR
.ad
.sp .6
.RS 4n
The amount of space used by a \fBrefreservation\fR set on this dataset, which
would be freed if the \fBrefreservation\fR was removed.
.RE

.sp
.ne 2
.na
\fB\fBusedbysnapshots\fR\fR
.ad
.sp .6
.RS 4n
The amount of space consumed by snapshots of this dataset. In particular, it is
the amount of space that would be freed if all of this dataset's snapshots were
destroyed. Note that this is not simply the sum of the snapshots' \fBused\fR
properties because space can be shared by multiple snapshots.
.RE

.sp
.ne 2
.na
\fB\fBuserused@\fR\fIuser\fR\fR
.ad
.sp .6
.RS 4n
The amount of space consumed by the specified user in this dataset. Space is
charged to the owner of each file, as displayed by \fBls\fR \fB-l\fR. The
amount of space charged is displayed by \fBdu\fR and \fBls\fR \fB-s\fR. See the
\fBzfs userspace\fR subcommand for more information.
.sp
Unprivileged users can access only their own space usage. The root user, or a
user who has been granted the \fBuserused\fR privilege with \fBzfs allow\fR,
can access everyone's usage.
.sp
The \fBuserused@\fR... properties are not displayed by \fBzfs get all\fR. The
user's name must be appended after the \fB@\fR symbol, using one of the
following forms:
.RS +4
.TP
.ie t \(bu
.el o
\fIPOSIX name\fR (for example, \fBjoe\fR)
.RE
.RS +4
.TP
.ie t \(bu
.el o
\fIPOSIX numeric ID\fR (for example, \fB789\fR)
.RE
.RS +4
.TP
.ie t \(bu
.el o
\fISID name\fR (for example, \fBjoe.smith@mydomain\fR)
.RE
.RS +4
.TP
.ie t \(bu
.el o
```



```
\fISID numeric ID\fR (for example, \fBS-1-123-456-789\fR)
.RE
.RE

.sp
.ne 2
.na
\fB\fBuserrefs\fR\fR
.ad
.sp .6
.RS 4n
This property is set to the number of user holds on this snapshot. User holds
are set by using the \fBzfs hold\fR command.
.RE

.sp
.ne 2
.na
\fB\fBgroupused@\fR\fIgroup\fR\fR
.ad
.sp .6
.RS 4n
The amount of space consumed by the specified group in this dataset. Space is
charged to the group of each file, as displayed by \fBls\fR \fB-l\fR. See the
\fBuserused@\fR\fIuser\fR property for more information.
.sp
Unprivileged users can only access their own groups' space usage. The root
user, or a user who has been granted the \fBgroupused\fR privilege with \fBzfs
allow\fR, can access all groups' usage.
.RE

.sp
.ne 2
.na
\fB\fBvolblocksize\fR=\fIblocksize\fR\fR
.ad
.sp .6
.RS 4n
For volumes, specifies the block size of the volume. The \fBblocksize\fR cannot
be changed once the volume has been written, so it should be set at volume
creation time. The default \fBblocksize\fR for volumes is 8 Kbytes. Any power
of 2 from 512 bytes to 128 Kbytes is valid.
.sp
This property can also be referred to by its shortened column name,
\fBvolblock\fR.
.RE

.sp
.ne 2
.na
\fB\fBwritten\fR\fR
.ad
.sp .6
.RS 4n
The amount of \fBreferenced\fR space written to this dataset since the
previous snapshot.
.RE

.sp
.ne 2
.na
\fB\fBwritten@\fR\fIsnapshot\fR\fR
.ad
.sp .6
.RS 4n
The amount of \fBreferenced\fR space written to this dataset since the
```

specified snapshot. This is the space that is referenced by this dataset
but was not referenced by the specified snapshot.
.sp
The \fIsnapshot\fR may be specified as a short snapshot name (just the part
after the \fB@\fR), in which case it will be interpreted as a snapshot in
the same filesystem as this dataset.
The \fIsnapshot\fR be a full snapshot name (\fIfilesystem\fR@\fIsnapshot\fR),
which for clones may be a snapshot in the origin's filesystem (or the origin
of the origin's filesystem, etc).
.RE

.sp
.LP
The following native properties can be used to change the behavior of a
\fBZFS\fR dataset.
.sp
.ne 2
.na
\fB\fBaclinherit\fR=\fBdiscard\fR | \fBnoallow\fR | \fBrestricted\fR |
\fBpassthrough\fR | \fBpassthrough-x\fR\fR
.ad
.sp .6
.RS 4n
Controls how \fBACL\fR entries are inherited when files and directories are
created. A file system with an \fBaclinherit\fR property of \fBdiscard\fR does
not inherit any \fBACL\fR entries. A file system with an \fBaclinherit\fR
property value of \fBnoallow\fR only inherits inheritable \fBACL\fR entries
that specify "deny" permissions. The property value \fBrestricted\fR (the
default) removes the \fBwrite_acl\fR and \fBwrite_owner\fR permissions when the
\fBACL\fR entry is inherited. A file system with an \fBaclinherit\fR property
value of \fBpassthrough\fR inherits all inheritable \fBACL\fR entries without
any modifications made to the \fBACL\fR entries when they are inherited. A file
system with an \fBaclinherit\fR property value of \fBpassthrough-x\fR has the
same meaning as \fBpassthrough\fR, except that the \fBowner@\fR, \fBgroup@\fR,
and \fBeveryone@\fR \fBACE\fRs inherit the execute permission only if the file
creation mode also requests the execute bit.
.sp
When the property value is set to \fBpassthrough\fR, files are created with a
mode determined by the inheritable \fBACE\fRs. If no inheritable \fBACE\fRs
exist that affect the mode, then the mode is set in accordance to the requested
mode from the application.
.RE

.sp
.ne 2
.na
\fB\fBaclmode\fR=\fBdiscard\fR | \fBgroupmask\fR | \fBpassthrough\fR\fR | \fBres
.ad
.sp .6
.RS 4n
Controls how an \fBACL\fR is modified during \fBchmod\fR(2). A file system with
an \fBaclmode\fR property of \fBdiscard\fR (the default) deletes all \fBACL\fR
entries that do not represent the mode of the file. An \fBaclmode\fR property
of \fBgroupmask\fR reduces permissions granted in all \fBALLOW\fR entries found
in the \fBACL\fR such that they are no greater than the group permissions
specified by \fBchmod\fR(2). A file system with an \fBaclmode\fR property of
\fBpassthrough\fR indicates that no changes are made to the \fBACL\fR other
than creating or updating the necessary \fBACL\fR entries to represent the new
mode of the file or directory. An \fBaclmode\fR property of \fBrestricted\fR
will cause the \fBchmod\fR(2) operation to return an error when used on any
file or directory which has a non-trivial \fBACL\fR whose entries can not be
represented by a mode. \fBchmod\fR(2) is required to change the set user ID,
set group ID, or sticky bits on a file or directory, as they do not have
equivalent \fBACL\fR entries. In order to use \fBchmod\fR(2) on a file or
directory with a non-trivial \fBACL\fR when \fBaclmode\fR is set to
\fBrestricted\fR, you must first remove all \fBACL\fR entries which do not

represent the current mode.
.RE

.sp
.ne 2
.na
\fB\fBatime\fR=\fBon\fR | \fBoff\fR\fR
.ad
.sp .6
.RS 4n
Controls whether the access time for files is updated when they are read.
Turning this property off avoids producing write traffic when reading files and
can result in significant performance gains, though it might confuse mailers
and other similar utilities. The default value is \fBon\fR.
.RE

.sp
.ne 2
.na
\fB\fBcanmount\fR=\fBon\fR | \fBoff\fR | \fBnoauto\fR\fR
.ad
.sp .6
.RS 4n
If this property is set to \fBoff\fR, the file system cannot be mounted, and is
ignored by \fBzfs mount -a\fR. Setting this property to \fBoff\fR is similar to
setting the \fBmountpoint\fR property to \fBnone\fR, except that the dataset
still has a normal \fBmountpoint\fR property, which can be inherited. Setting
this property to \fBoff\fR allows datasets to be used solely as a mechanism to
inherit properties. One example of setting \fBcanmount=\fR\fBoff\fR is to have
two datasets with the same \fBmountpoint\fR, so that the children of both
datasets appear in the same directory, but might have different inherited
characteristics.
.sp
When the \fBnoauto\fR option is set, a dataset can only be mounted and
unmounted explicitly. The dataset is not mounted automatically when the dataset
is created or imported, nor is it mounted by the \fBzfs mount -a\fR command or
unmounted by the \fBzfs unmount -a\fR command.
.sp
This property is not inherited.
.RE

.sp
.ne 2
.na
\fB\fBchecksum\fR=\fBon\fR | \fBoff\fR | \fBfletcher2\fR | \fBfletcher4\fR |
\fBsha256\fR\fR
.ad
.sp .6
.RS 4n
Controls the checksum used to verify data integrity. The default value is
\fBon\fR, which automatically selects an appropriate algorithm (currently,
\fBfletcher4\fR, but this may change in future releases). The value \fBoff\fR
disables integrity checking on user data. Disabling checksums is \fBNOT\fR a
recommended practice.
.sp
Changing this property affects only newly-written data.
.RE

.sp
.ne 2
.na
\fB\fBcompression\fR=\fBon\fR | \fBoff\fR | \fBlzjb\fR | \fBgzip\fR |
\fBgzip-\fR\fIN\fR | \fBzle\fR\fR | \fBlz4\fR | \fBlz4hc\fR
\fIgzip-\fR\fIN\fR | \fIzle\fR\fR
.ad
.sp .6

```
 919 .RS 4n
 920 Controls the compression algorithm used for this dataset. The \fBlzjb\fR
 921 compression algorithm is optimized for performance while providing decent data
 922 compression. Setting compression to \fBon\fR uses the \fBlzjb\fR compression
 923 algorithm. The \fBgzip\fR compression algorithm uses the same compression as
 924 the \fBgzip\fR(1) command. You can specify the \fBgzip\fR level by using the
 925 value \fBgzip-\fR\fIN\fR where \fIN\fR is an integer from 1 (fastest) to 9
 926 (best compression ratio). Currently, \fBgzip\fR is equivalent to \fBgzip-6\fR
 927 (which is also the default for \fBgzip\fR(1)). The \fBzle\fR compression
 928 algorithm compresses runs of zeros.
 929 .sp
 930 The \fBlz4\fR compression algorithm is a high-performance replacement
 931 for the \fBlzjb\fR algorithm. It features significantly faster
 932 compression and decompression, as well as a moderately higher
 933 compression ratio than \fBlzjb\fR, but can only be used on pools with
 934 the \fBlz4_compress\fR feature set to \fIenabled\fR. See
 935 \fBzpool-features\fR(5) for details on ZFS feature flags and the
 936 \fBlz4_compress\fR feature. The \fBlz4hc\fR algorithm is a
 937 high-compression variant of \fBlz4\fR. It provides compression ratios
 938 and compression speeds comparable to gzip, but much faster decompression
 939 (same as \fBlz4\fR).
 940 .sp
 941 This property can also be referred to by its shortened column name
 942 \fBcompress\fR. Changing this property affects only newly-written data.
 943 .RE

 945 .sp
 946 .ne 2
 947 .na
 948 \fB\fBcopies\fR=\fB1\fR | \fB2\fR | \fB3\fR\fR
 949 .ad
 950 .sp .6
 951 .RS 4n
 952 Controls the number of copies of data stored for this dataset. These copies are
 953 in addition to any redundancy provided by the pool, for example, mirroring or
 954 RAID-Z. The copies are stored on different disks, if possible. The space used
 955 by multiple copies is charged to the associated file and dataset, changing the
 956 \fBused\fR property and counting against quotas and reservations.
 957 .sp
 958 Changing this property only affects newly-written data. Therefore, set this
 959 property at file system creation time by using the \fB-o\fR
 960 \fBcopies=\fR\fIN\fR option.
 961 .RE

 963 .sp
 964 .ne 2
 965 .na
 966 \fB\fBdevices\fR=\fBon\fR | \fBoff\fR\fR
 967 .ad
 968 .sp .6
 969 .RS 4n
 970 Controls whether device nodes can be opened on this file system. The default
 971 value is \fBon\fR.
 972 .RE

 974 .sp
 975 .ne 2
 976 .na
 977 \fB\fBexec\fR=\fBon\fR | \fBoff\fR\fR
 978 .ad
 979 .sp .6
 980 .RS 4n
 981 Controls whether processes can be executed from within this file system. The
 982 default value is \fBon\fR.
 983 .RE
```



```
 985 .sp
 986 .ne 2
 987 .na
 988 \fB\fBmountpoint\fR=\fIpath\fR | \fBnone\fR | \fBlegacy\fR\fR
 989 .ad
 990 .sp .6
 991 .RS 4n
 992 Controls the mount point used for this file system. See the "Mount Points"
 993 section for more information on how this property is used.
 994 .sp
 995 When the \fBmountpoint\fR property is changed for a file system, the file
 996 system and any children that inherit the mount point are unmounted. If the new
 997 value is \fBlegacy\fR, then they remain unmounted. Otherwise, they are
 998 automatically remounted in the new location if the property was previously
 999 \fBlegacy\fR or \fBnone\fR, or if they were mounted before the property was
1000 changed. In addition, any shared file systems are unshared and shared in the
1001 new location.
1002 .RE

1004 .sp
1005 .ne 2
1006 .na
1007 \fB\fBnbmand\fR=\fBon\fR | \fBoff\fR\fR
1008 .ad
1009 .sp .6
1010 .RS 4n
1011 Controls whether the file system should be mounted with \fBnbmand\fR (Non
1012 Blocking mandatory locks). This is used for \fBCIFS\fR clients. Changes to this
1013 property only take effect when the file system is umounted and remounted. See
1014 \fBmount\fR(1M) for more information on \fBnbmand\fR mounts.
1015 .RE

1017 .sp
1018 .ne 2
1019 .na
1020 \fB\fBprimarycache\fR=\fBall\fR | \fBnone\fR | \fBmetadata\fR\fR
1021 .ad
1022 .sp .6
1023 .RS 4n
1024 Controls what is cached in the primary cache (ARC). If this property is set to
1025 \fBall\fR, then both user data and metadata is cached. If this property is set
1026 to \fBnone\fR, then neither user data nor metadata is cached. If this property
1027 is set to \fBmetadata\fR, then only metadata is cached. The default value is
1028 \fBall\fR.
1029 .RE

1031 .sp
1032 .ne 2
1033 .na
1034 \fB\fBquota\fR=\fIsize\fR | \fBnone\fR\fR
1035 .ad
1036 .sp .6
1037 .RS 4n
1038 Limits the amount of space a dataset and its descendents can consume. This
1039 property enforces a hard limit on the amount of space used. This includes all
1040 space consumed by descendents, including file systems and snapshots. Setting a
1041 quota on a descendent of a dataset that already has a quota does not override
1042 the ancestor’s quota, but rather imposes an additional limit.
1043 .sp
1044 Quotas cannot be set on volumes, as the \fBvolsize\fR property acts as an
1045 implicit quota.
1046 .RE

1048 .sp
1049 .ne 2
1050 .na
```

\fB\fBuserquota@\fR\fIuser\fR=\fIsize\fR | \fBnone\fR\fR
.ad
.sp .6
.RS 4n
Limits the amount of space consumed by the specified user. User space consumption is identified by the \fBuserspace@\fR\fIuser\fR property.
.sp
Enforcement of user quotas may be delayed by several seconds. This delay means that a user might exceed their quota before the system notices that they are over quota and begins to refuse additional writes with the \fBEDQUOT\fR error message . See the \fBzfs userspace\fR subcommand for more information.
.sp
Unprivileged users can only access their own groups' space usage. The root user, or a user who has been granted the \fBuserquota\fR privilege with \fBzfs allow\fR, can get and set everyone's quota.
.sp
This property is not available on volumes, on file systems before version 4, or on pools before version 15. The \fBuserquota@\fR... properties are not displayed by \fBzfs get all\fR. The user's name must be appended after the \fB@\fR symbol, using one of the following forms:
.RS +4
.TP
.ie t \(bu
.el o
\fIPOSIX name\fR (for example, \fBjoe\fR)
.RE
.RS +4
.TP
.ie t \(bu
.el o
\fIPOSIX numeric ID\fR (for example, \fB789\fR)
.RE
.RS +4
.TP
.ie t \(bu
.el o
\fISID name\fR (for example, \fBjoe.smith@mydomain\fR)
.RE
.RS +4
.TP
.ie t \(bu
.el o
\fISID numeric ID\fR (for example, \fBS-1-123-456-789\fR)
.RE
.RE

.sp
.ne 2
.na
\fB\fBgroupquota@\fR\fIgroup\fR=\fIsize\fR | \fBnone\fR\fR
.ad
.sp .6
.RS 4n
Limits the amount of space consumed by the specified group. Group space consumption is identified by the \fBuserquota@\fR\fIuser\fR property.
.sp
Unprivileged users can access only their own groups' space usage. The root user, or a user who has been granted the \fBgroupquota\fR privilege with \fBzfs allow\fR, can get and set all groups' quotas.
.RE

.sp
.ne 2
.na
\fB\fBreadonly\fR=\fBon\fR | \fBoff\fR\fR
.ad

.sp .6
.RS 4n
Controls whether this dataset can be modified. The default value is \fBoff\fR.
.sp
This property can also be referred to by its shortened column name, \fBrdonly\fR.
.RE

.sp
.ne 2
.na
\fB\fBrecordsize\fR=\fIsize\fR\fR
.ad
.sp .6
.RS 4n
Specifies a suggested block size for files in the file system. This property is designed solely for use with database workloads that access files in fixed-size records. \fBZFS\fR automatically tunes block sizes according to internal algorithms optimized for typical access patterns.
.sp
For databases that create very large files but access them in small random chunks, these algorithms may be suboptimal. Specifying a \fBrecordsize\fR greater than or equal to the record size of the database can result in significant performance gains. Use of this property for general purpose file systems is strongly discouraged, and may adversely affect performance.
.sp
The size specified must be a power of two greater than or equal to 512 and less than or equal to 128 Kbytes.
.sp
Changing the file system's \fBrecordsize\fR affects only files created afterward; existing files are unaffected.
.sp
This property can also be referred to by its shortened column name, \fBrecsize\fR.
.RE

.sp
.ne 2
.na
\fB\fBrefquota\fR=\fIsize\fR | \fBnone\fR\fR
.ad
.sp .6
.RS 4n
Limits the amount of space a dataset can consume. This property enforces a hard limit on the amount of space used. This hard limit does not include space used by descendents, including file systems and snapshots.
.RE

.sp
.ne 2
.na
\fB\fBrefreservation\fR=\fIsize\fR | \fBnone\fR\fR
.ad
.sp .6
.RS 4n
The minimum amount of space guaranteed to a dataset, not including its descendents. When the amount of space used is below this value, the dataset is treated as if it were taking up the amount of space specified by \fBrefreservation\fR. The \fBrefreservation\fR reservation is accounted for in the parent datasets' space used, and counts against the parent datasets' quotas and reservations.
.sp
If \fBrefreservation\fR is set, a snapshot is only allowed if there is enough free pool space outside of this reservation to accommodate the current number of "referenced" bytes in the dataset.
.sp

```
1183 This property can also be referred to by its shortened column name,
1184 \fBrefreserv\fR.
1185 .RE

1187 .sp
1188 .ne 2
1189 .na
1190 \fB\fBreservation\fR=\fIsize\fR | \fBnone\fR\fR
1191 .ad
1192 .sp .6
1193 .RS 4n
1194 The minimum amount of space guaranteed to a dataset and its descendents. When
1195 the amount of space used is below this value, the dataset is treated as if it
1196 were taking up the amount of space specified by its reservation. Reservations
1197 are accounted for in the parent datasets' space used, and count against the
1198 parent datasets' quotas and reservations.
1199 .sp
1200 This property can also be referred to by its shortened column name,
1201 \fBreserv\fR.
1202 .RE

1204 .sp
1205 .ne 2
1206 .na
1207 \fB\fBsecondarycache\fR=\fBall\fR | \fBnone\fR | \fBmetadata\fR\fR
1208 .ad
1209 .sp .6
1210 .RS 4n
1211 Controls what is cached in the secondary cache (L2ARC). If this property is set
1212 to \fBall\fR, then both user data and metadata is cached. If this property is
1213 set to \fBnone\fR, then neither user data nor metadata is cached. If this
1214 property is set to \fBmetadata\fR, then only metadata is cached. The default
1215 value is \fBall\fR.
1216 .RE

1218 .sp
1219 .ne 2
1220 .na
1221 \fB\fBsetuid\fR=\fBon\fR | \fBoff\fR\fR
1222 .ad
1223 .sp .6
1224 .RS 4n
1225 Controls whether the set-\fBUID\fR bit is respected for the file system. The
1226 default value is \fBon\fR.
1227 .RE

1229 .sp
1230 .ne 2
1231 .na
1232 \fB\fBshareiscsi\fR=\fBon\fR | \fBoff\fR\fR
1233 .ad
1234 .sp .6
1235 .RS 4n
1236 Like the \fBsharenfs\fR property, \fBshareiscsi\fR indicates whether a
1237 \fBZFS\fR volume is exported as an \fBiSCSI\fR target. The acceptable values
1238 for this property are \fBon\fR, \fBoff\fR, and \fBtype=disk\fR. The default
1239 value is \fBoff\fR. In the future, other target types might be supported. For
1240 example, \fBtape\fR.
1241 .sp
1242 You might want to set \fBshareiscsi=on\fR for a file system so that all
1243 \fBZFS\fR volumes within the file system are shared by default. However,
1244 setting this property on a file system has no direct effect.
1245 .RE

1247 .sp
1248 .ne 2
```



```
1249 .na
1250 \fB\fBsharesmb\fR=\fBon\fR | \fBoff\fR | \fIopts\fR\fR
1251 .ad
1252 .sp .6
1253 .RS 4n
1254 Controls whether the file system is shared by using the Solaris \fBCIFS\fR
1255 service, and what options are to be used. A file system with the \fBsharesmb\fR
1256 property set to \fBoff\fR is managed through traditional tools such as
1257 \fBsharemgr\fR(1M). Otherwise, the file system is automatically shared and
1258 unshared with the \fBzfs share\fR and \fBzfs unshare\fR commands. If the
1259 property is set to \fBon\fR, the \fBsharemgr\fR(1M) command is invoked with no
1260 options. Otherwise, the \fBsharemgr\fR(1M) command is invoked with options
1261 equivalent to the contents of this property.
1262 .sp
1263 Because \fBSMB\fR shares requires a resource name, a unique resource name is
1264 constructed from the dataset name. The constructed name is a copy of the
1265 dataset name except that the characters in the dataset name, which would be
1266 illegal in the resource name, are replaced with underscore (\fB_\fR)
1267 characters. A pseudo property "name" is also supported that allows you to
1268 replace the data set name with a specified name. The specified name is then
1269 used to replace the prefix dataset in the case of inheritance. For example, if
1270 the dataset \fBdata/home/john\fR is set to \fBname=john\fR, then
1271 \fBdata/home/john\fR has a resource name of \fBjohn\fR. If a child dataset of
1272 \fBdata/home/john/backups\fR, it has a resource name of \fBjohn_backups\fR.
1273 .sp
1274 When SMB shares are created, the SMB share name appears as an entry in the
1275 \fB\&.zfs/shares\fR directory. You can use the \fBls\fR or \fBchmod\fR command
1276 to display the share-level ACLs on the entries in this directory.
1277 .sp
1278 When the \fBsharesmb\fR property is changed for a dataset, the dataset and any
1279 children inheriting the property are re-shared with the new options, only if
1280 the property was previously set to \fBoff\fR, or if they were shared before the
1281 property was changed. If the new property is set to \fBoff\fR, the file systems
1282 are unshared.
1283 .RE

1285 .sp
1286 .ne 2
1287 .na
1288 \fB\fBsharenfs\fR=\fBon\fR | \fBoff\fR | \fIopts\fR\fR
1289 .ad
1290 .sp .6
1291 .RS 4n
1292 Controls whether the file system is shared via \fBNFS\fR, and what options are
1293 used. A file system with a \fBsharenfs\fR property of \fBoff\fR is managed
1294 through traditional tools such as \fBshare\fR(1M), \fBunshare\fR(1M), and
1295 \fBdfstab\fR(4). Otherwise, the file system is automatically shared and
1296 unshared with the \fBzfs share\fR and \fBzfs unshare\fR commands. If the
1297 property is set to \fBon\fR, the \fBshare\fR(1M) command is invoked with no
1298 options. Otherwise, the \fBshare\fR(1M) command is invoked with options
1299 equivalent to the contents of this property.
1300 .sp
1301 When the \fBsharenfs\fR property is changed for a dataset, the dataset and any
1302 children inheriting the property are re-shared with the new options, only if
1303 the property was previously \fBoff\fR, or if they were shared before the
1304 property was changed. If the new property is \fBoff\fR, the file systems are
1305 unshared.
1306 .RE

1308 .sp
1309 .ne 2
1310 .na
1311 \fB\fBlogbias\fR = \fBlatency\fR | \fBthroughput\fR\fR
1312 .ad
1313 .sp .6
1314 .RS 4n
```

Provide a hint to ZFS about handling of synchronous requests in this dataset. If \fBlogbias\fR is set to \fBlatency\fR (the default), ZFS will use pool log devices (if configured) to handle the requests at low latency. If \fBlogbias\fR is set to \fBthroughput\fR, ZFS will not use configured pool log devices. ZFS will instead optimize synchronous operations for global pool throughput and efficient use of resources.
.RE

.sp
.ne 2
.na
\fB\fBsnapdir\fR=\fBhidden\fR | \fBvisible\fR\fR
.ad
.sp .6
.RS 4n
Controls whether the \fB\&.zfs\fR directory is hidden or visible in the root of the file system as discussed in the "Snapshots" section. The default value is \fBhidden\fR.
.RE

.sp
.ne 2
.na
\fB\fBsync\fR=\fBdefault\fR | \fBalways\fR | \fBdisabled\fR\fR
.ad
.sp .6
.RS 4n
Controls the behavior of synchronous requests (e.g. fsync, O_DSYNC). \fBdefault\fR is the POSIX specified behavior of ensuring all synchronous requests are written to stable storage and all devices are flushed to ensure data is not cached by device controllers (this is the default). \fBalways\fR causes every file system transaction to be written and flushed before its system call returns. This has a large performance penalty. \fBdisabled\fR disables synchronous requests. File system transactions are only committed to stable storage periodically. This option will give the highest performance. However, it is very dangerous as ZFS would be ignoring the synchronous transaction demands of applications such as databases or NFS. Administrators should only use this option when the risks are understood.
.RE

.sp
.ne 2
.na
\fB\fBversion\fR=\fB1\fR | \fB2\fR | \fBcurrent\fR\fR
.ad
.sp .6
.RS 4n
The on-disk version of this file system, which is independent of the pool version. This property can only be set to later supported versions. See the \fBzfs upgrade\fR command.
.RE

.sp
.ne 2
.na
\fB\fBvolsize\fR=\fIsize\fR\fR
.ad
.sp .6
.RS 4n
For volumes, specifies the logical size of the volume. By default, creating a volume establishes a reservation of equal size. For storage pools with a version number of 9 or higher, a \fBrefreservation\fR is set instead. Any changes to \fBvolsize\fR are reflected in an equivalent change to the reservation (or \fBrefreservation\fR). The \fBvolsize\fR can only be set to a multiple of \fBvolblocksize\fR, and cannot be zero.
.sp

The reservation is kept equal to the volume’s logical size to prevent unexpected behavior for consumers. Without the reservation, the volume could run out of space, resulting in undefined behavior or data corruption, depending on how the volume is used. These effects can also occur when the volume size is changed while it is in use (particularly when shrinking the size). Extreme care should be used when adjusting the volume size.
.sp
Though not recommended, a "sparse volume" (also known as "thin provisioning") can be created by specifying the \fB-s\fR option to the \fBzfs create -V\fR command, or by changing the reservation after the volume has been created. A "sparse volume" is a volume where the reservation is less then the volume size. Consequently, writes to a sparse volume can fail with \fBENOSPC\fR when the pool is low on space. For a sparse volume, changes to \fBvolsize\fR are not reflected in the reservation.
.RE

.sp
.ne 2
.na
\fB\fBvscan\fR=\fBon\fR | \fBoff\fR\fR
.ad
.sp .6
.RS 4n
Controls whether regular files should be scanned for viruses when a file is opened and closed. In addition to enabling this property, the virus scan service must also be enabled for virus scanning to occur. The default value is \fBoff\fR.
.RE

.sp
.ne 2
.na
\fB\fBxattr\fR=\fBon\fR | \fBoff\fR\fR
.ad
.sp .6
.RS 4n
Controls whether extended attributes are enabled for this file system. The default value is \fBon\fR.
.RE

.sp
.ne 2
.na
\fB\fBzoned\fR=\fBon\fR | \fBoff\fR\fR
.ad
.sp .6
.RS 4n
Controls whether the dataset is managed from a non-global zone. See the "Zones" section for more information. The default value is \fBoff\fR.
.RE

.sp
.LP
The following three properties cannot be changed after the file system is created, and therefore, should be set when the file system is created. If the properties are not set with the \fBzfs create\fR or \fBzpool create\fR commands, these properties are inherited from the parent dataset. If the parent dataset lacks these properties due to having been created prior to these features being supported, the new file system will have the default values for these properties.
.sp
.ne 2
.na
\fB\fBcasesensitivity\fR=\fBsensitive\fR | \fBinsensitive\fR | \fBmixed\fR\fR
.ad
.sp .6

.RS 4n
Indicates whether the file name matching algorithm used by the file system should be case-sensitive, case-insensitive, or allow a combination of both styles of matching. The default value for the \fBcasesensitivity\fR property is \fBsensitive\fR. Traditionally, UNIX and POSIX file systems have case-sensitive file names.
.sp
The \fBmixed\fR value for the \fBcasesensitivity\fR property indicates that the file system can support requests for both case-sensitive and case-insensitive matching behavior. Currently, case-insensitive matching behavior on a file system that supports mixed behavior is limited to the Solaris CIFS server product. For more information about the \fBmixed\fR value behavior, see the \fISolaris ZFS Administration Guide\fR.
.RE

.sp
.ne 2
.na
\fB\fBnormalization\fR = \fBnone\fR | \fBformC\fR | \fBformD\fR | \fBformKC\fR | \fBformKD\fR\fR
.ad
.sp .6
.RS 4n
Indicates whether the file system should perform a \fBunicode\fR normalization of file names whenever two file names are compared, and which normalization algorithm should be used. File names are always stored unmodified, names are normalized as part of any comparison process. If this property is set to a legal value other than \fBnone\fR, and the \fButf8only\fR property was left unspecified, the \fButf8only\fR property is automatically set to \fBon\fR. The default value of the \fBnormalization\fR property is \fBnone\fR. This property cannot be changed after the file system is created.
.RE

.sp
.ne 2
.na
\fB\fButf8only\fR=\fBon\fR | \fBoff\fR\fR
.ad
.sp .6
.RS 4n
Indicates whether the file system should reject file names that include characters that are not present in the \fBUTF-8\fR character code set. If this property is explicitly set to \fBoff\fR, the normalization property must either not be explicitly set or be set to \fBnone\fR. The default value for the \fButf8only\fR property is \fBoff\fR. This property cannot be changed after the file system is created.
.RE

.sp
.LP
The \fBcasesensitivity\fR, \fBnormalization\fR, and \fButf8only\fR properties are also new permissions that can be assigned to non-privileged users by using the \fBZFS\fR delegated administration feature.
.SS "Temporary Mount Point Properties"
.sp
.LP
When a file system is mounted, either through \fBmount\fR(1M) for legacy mounts or the \fBzfs mount\fR command for normal file systems, its mount options are set according to its properties. The correlation between properties and mount options is as follows:
.sp
.in +2
.nf
    PROPERTY   MOUNT OPTION
     devices   devices/nodevices
     exec   exec/noexec

     readonly   ro/rw
     setuid   setuid/nosetuid
     xattr   xattr/noxattr
.fi
.in -2
.sp

.sp
.LP
In addition, these options can be set on a per-mount basis using the \fB-o\fR option, without affecting the property that is stored on disk. The values specified on the command line override the values stored in the dataset. The \fB-nosuid\fR option is an alias for \fBnodevices,nosetuid\fR. These properties are reported as "temporary" by the \fBzfs get\fR command. If the properties are changed while the dataset is mounted, the new setting overrides any temporary settings.
.SS "User Properties"
.sp
.LP
In addition to the standard native properties, \fBZFS\fR supports arbitrary user properties. User properties have no effect on \fBZFS\fR behavior, but applications or administrators can use them to annotate datasets (file systems, volumes, and snapshots).
.sp
.LP
User property names must contain a colon (\fB:\fR) character to distinguish them from native properties. They may contain lowercase letters, numbers, and the following punctuation characters: colon (\fB:\fR), dash (\fB-\fR), period (\fB\&.\fR), and underscore (\fB_\fR). The expected convention is that the property name is divided into two portions such as \fImodule\fR\fB:\fR\fIproperty\fR, but this namespace is not enforced by \fBZFS\fR. User property names can be at most 256 characters, and cannot begin with a dash (\fB-\fR).
.sp
.LP
When making programmatic use of user properties, it is strongly suggested to use a reversed \fBDNS\fR domain name for the \fImodule\fR component of property names to reduce the chance that two independently-developed packages use the same property name for different purposes. Property names beginning with \fBcom.sun\fR. are reserved for use by Sun Microsystems.
.sp
.LP
The values of user properties are arbitrary strings, are always inherited, and are never validated. All of the commands that operate on properties (\fBzfs list\fR, \fBzfs get\fR, \fBzfs set\fR, and so forth) can be used to manipulate both native properties and user properties. Use the \fBzfs inherit\fR command to clear a user property . If the property is not defined in any parent dataset, it is removed entirely. Property values are limited to 1024 characters.
.SS "ZFS Volumes as Swap or Dump Devices"
.sp
.LP
During an initial installation a swap device and dump device are created on \fBZFS\fR volumes in the \fBZFS\fR root pool. By default, the swap area size is based on 1/2 the size of physical memory up to 2 Gbytes. The size of the dump device depends on the kernel’s requirements at installation time. Separate \fBZFS\fR volumes must be used for the swap area and dump devices. Do not swap to a file on a \fBZFS\fR file system. A \fBZFS\fR swap file configuration is not supported.
.sp
.LP
If you need to change your swap area or dump device after the system is installed or upgraded, use the \fBswap\fR(1M) and \fBdumpadm\fR(1M) commands. If you need to change the size of your swap area or dump device, see the \fISolaris ZFS Administration Guide\fR.
.SH SUBCOMMANDS

```
.sp
.LP
All subcommands that modify state are logged persistently to the pool in their
original form.
.sp
.ne 2
.na
\fB\fBzfs ?\fR\fR
.ad
.sp .6
.RS 4n
Displays a help message.
.RE

.sp
.ne 2
.na
\fB\fBzfs create\fR [\fB-p\fR] [\fB-o\fR \fIproperty\fR=\fIvalue\fR] ...
\fIfilesystem\fR\fR
.ad
.sp .6
.RS 4n
Creates a new \fBZFS\fR file system. The file system is automatically mounted
according to the \fBmountpoint\fR property inherited from the parent.
.sp
.ne 2
.na
\fB\fB-p\fR\fR
.ad
.sp .6
.RS 4n
Creates all the non-existing parent datasets. Datasets created in this manner
are automatically mounted according to the \fBmountpoint\fR property inherited
from their parent. Any property specified on the command line using the
\fB-o\fR option is ignored. If the target filesystem already exists, the
operation completes successfully.
.RE

.sp
.ne 2
.na
\fB\fB-o\fR \fIproperty\fR=\fIvalue\fR\fR
.ad
.sp .6
.RS 4n
Sets the specified property as if the command \fBzfs set\fR
\fIproperty\fR=\fIvalue\fR was invoked at the same time the dataset was
created. Any editable \fBZFS\fR property can also be set at creation time.
Multiple \fB-o\fR options can be specified. An error results if the same
property is specified in multiple \fB-o\fR options.
.RE

.RE

.sp
.ne 2
.na
\fB\fBzfs create\fR [\fB-ps\fR] [\fB-b\fR \fIblocksize\fR] [\fB-o\fR
\fIproperty\fR=\fIvalue\fR] ... \fB-V\fR \fIsize\fR \fIvolume\fR\fR
.ad
.sp .6
.RS 4n
Creates a volume of the given size. The volume is exported as a block device in
\fB/dev/zvol/{dsk,rdsk}/\fR\fIpath\fR, where \fIpath\fR is the name of the
volume in the \fBZFS\fR namespace. The size represents the logical size as
exported by the device. By default, a reservation of equal size is created.
```



```
.sp
\fIsize\fR is automatically rounded up to the nearest 128 Kbytes to ensure that
the volume has an integral number of blocks regardless of \fIblocksize\fR.
.sp
.ne 2
.na
\fB\fB-p\fR\fR
.ad
.sp .6
.RS 4n
Creates all the non-existing parent datasets. Datasets created in this manner
are automatically mounted according to the \fBmountpoint\fR property inherited
from their parent. Any property specified on the command line using the
\fB-o\fR option is ignored. If the target filesystem already exists, the
operation completes successfully.
.RE

.sp
.ne 2
.na
\fB\fB-s\fR\fR
.ad
.sp .6
.RS 4n
Creates a sparse volume with no reservation. See \fBvolsize\fR in the Native
Properties section for more information about sparse volumes.
.RE

.sp
.ne 2
.na
\fB\fB-o\fR \fIproperty\fR=\fIvalue\fR\fR
.ad
.sp .6
.RS 4n
Sets the specified property as if the \fBzfs set\fR \fIproperty\fR=\fIvalue\fR
command was invoked at the same time the dataset was created. Any editable
\fBZFS\fR property can also be set at creation time. Multiple \fB-o\fR options
can be specified. An error results if the same property is specified in
multiple \fB-o\fR options.
.RE

.sp
.ne 2
.na
\fB\fB-b\fR \fIblocksize\fR\fR
.ad
.sp .6
.RS 4n
Equivalent to \fB-o\fR \fBvolblocksize\fR=\fIblocksize\fR. If this option is
specified in conjunction with \fB-o\fR \fBvolblocksize\fR, the resulting
behavior is undefined.
.RE

.RE

.sp
.ne 2
.na
\fBzfs destroy\fR [\fB-fnpRrv\fR] \fIfilesystem\fR|\fIvolume\fR
.ad
.sp .6
.RS 4n
Destroys the given dataset. By default, the command unshares any file systems
that are currently shared, unmounts any file systems that are currently
mounted, and refuses to destroy a dataset that has active dependents (children
```

```
1711 or clones).
1712 .sp
1713 .ne 2
1714 .na
1715 \fB\fB-r\fR\fR
1716 .ad
1717 .sp .6
1718 .RS 4n
1719 Recursively destroy all children.
1720 .RE

1722 .sp
1723 .ne 2
1724 .na
1725 \fB\fB-R\fR\fR
1726 .ad
1727 .sp .6
1728 .RS 4n
1729 Recursively destroy all dependents, including cloned file systems outside the
1730 target hierarchy.
1731 .RE

1733 .sp
1734 .ne 2
1735 .na
1736 \fB\fB-f\fR\fR
1737 .ad
1738 .sp .6
1739 .RS 4n
1740 Force an unmount of any file systems using the \fBunmount -f\fR command. This
1741 option has no effect on non-file systems or unmounted file systems.
1742 .RE

1744 .sp
1745 .ne 2
1746 .na
1747 \fB\fB-n\fR\fR
1748 .ad
1749 .sp .6
1750 .RS 4n
1751 Do a dry-run ("No-op") deletion.  No data will be deleted.  This is
1752 useful in conjunction with the \fB-v\fR or \fB-p\fR flags to determine what
1753 data would be deleted.
1754 .RE

1756 .sp
1757 .ne 2
1758 .na
1759 \fB\fB-p\fR\fR
1760 .ad
1761 .sp .6
1762 .RS 4n
1763 Print machine-parsable verbose information about the deleted data.
1764 .RE

1766 .sp
1767 .ne 2
1768 .na
1769 \fB\fB-v\fR\fR
1770 .ad
1771 .sp .6
1772 .RS 4n
1773 Print verbose information about the deleted data.
1774 .RE
1775 .sp
1776 Extreme care should be taken when applying either the \fB-r\fR or the \fB-R\fR
```



```
1777 options, as they can destroy large portions of a pool and cause unexpected
1778 behavior for mounted file systems in use.
1779 .RE

1781 .sp
1782 .ne 2
1783 .na
1784 \fBzfs destroy\fR [\fB-dnpRrv\fR] \fIfilesystem\fR|\fIvolume\fR@\fIsnap\fR[%\fIs
1785 .ad
1786 .sp .6
1787 .RS 4n
1788 The given snapshots are destroyed immediately if and only if the \fBzfs
1789 destroy\fR command without the \fB-d\fR option would have destroyed it. Such
1790 immediate destruction would occur, for example, if the snapshot had no clones
1791 and the user-initiated reference count were zero.
1792 .sp
1793 If a snapshot does not qualify for immediate destruction, it is marked for
1794 deferred deletion. In this state, it exists as a usable, visible snapshot until
1795 both of the preconditions listed above are met, at which point it is destroyed.
1796 .sp
1797 An inclusive range of snapshots may be specified by separating the
1798 first and last snapshots with a percent sign.
1799 The first and/or last snapshots may be left blank, in which case the
1800 filesystem’s oldest or newest snapshot will be implied.
1801 .sp
1802 Multiple snapshots
1803 (or ranges of snapshots) of the same filesystem or volume may be specified
1804 in a comma-separated list of snapshots.
1805 Only the snapshot’s short name (the
1806 part after the \fB@\fR) should be specified when using a range or
1807 comma-separated list to identify multiple snapshots.
1808 .sp
1809 .ne 2
1810 .na
1811 \fB\fB-d\fR\fR
1812 .ad
1813 .sp .6
1814 .RS 4n
1815 Defer snapshot deletion.
1816 .RE

1818 .sp
1819 .ne 2
1820 .na
1821 \fB\fB-r\fR\fR
1822 .ad
1823 .sp .6
1824 .RS 4n
1825 Destroy (or mark for deferred deletion) all snapshots with this name in
1826 descendent file systems.
1827 .RE

1829 .sp
1830 .ne 2
1831 .na
1832 \fB\fB-R\fR\fR
1833 .ad
1834 .sp .6
1835 .RS 4n
1836 Recursively destroy all dependents.
1837 .RE

1839 .sp
1840 .ne 2
1841 .na
1842 \fB\fB-n\fR\fR
```

```
.ad
.sp .6
.RS 4n
Do a dry-run ("No-op") deletion.  No data will be deleted.  This is
useful in conjunction with the \fB-v\fR or \fB-p\fR flags to determine what
data would be deleted.
.RE

.sp
.ne 2
.na
\fB\fB-p\fR\fR
.ad
.sp .6
.RS 4n
Print machine-parsable verbose information about the deleted data.
.RE

.sp
.ne 2
.na
\fB\fB-v\fR\fR
.ad
.sp .6
.RS 4n
Print verbose information about the deleted data.
.RE

.sp
Extreme care should be taken when applying either the \fB-r\fR or the \fB-f\fR
options, as they can destroy large portions of a pool and cause unexpected
behavior for mounted file systems in use.
.RE

.sp
.ne 2
.na
\fB\fBzfs snapshot\fR [\fB-r\fR] [\fB-o\fR \fIproperty\fR=\fIvalue\fR] ...
\fIfilesystem@snapname\fR|\fIvolume@snapname\fR\fR...
.ad
.sp .6
.RS 4n
Creates snapshots with the given names. All previous modifications by
successful system calls to the file system are part of the snapshots.
Snapshots are taken atomically, so that all snapshots correspond to the same
moment in time. See the "Snapshots" section for details.
.sp
.ne 2
.na
\fB\fB-r\fR\fR
.ad
.sp .6
.RS 4n
Recursively create snapshots of all descendent datasets
.RE

.sp
.ne 2
.na
\fB\fB-o\fR \fIproperty\fR=\fIvalue\fR\fR
.ad
.sp .6
.RS 4n
Sets the specified property; see \fBzfs create\fR for details.
.RE
```



```
.RE

.sp
.ne 2
.na
\fB\fBzfs rollback\fR [\fB-rRf\fR] \fIsnapshot\fR\fR
.ad
.sp .6
.RS 4n
Roll back the given dataset to a previous snapshot. When a dataset is rolled
back, all data that has changed since the snapshot is discarded, and the
dataset reverts to the state at the time of the snapshot. By default, the
command refuses to roll back to a snapshot other than the most recent one. In
order to do so, all intermediate snapshots must be destroyed by specifying the
\fB-r\fR option.
.sp
The \fB-rR\fR options do not recursively destroy the child snapshots of a
recursive snapshot. Only the top-level recursive snapshot is destroyed by
either of these options. To completely roll back a recursive snapshot, you must
rollback the individual child snapshots.
.sp
.ne 2
.na
\fB\fB-r\fR\fR
.ad
.sp .6
.RS 4n
Recursively destroy any snapshots more recent than the one specified.
.RE

.sp
.ne 2
.na
\fB\fB-R\fR\fR
.ad
.sp .6
.RS 4n
Recursively destroy any more recent snapshots, as well as any clones of those
snapshots.
.RE

.sp
.ne 2
.na
\fB\fB-f\fR\fR
.ad
.sp .6
.RS 4n
Used with the \fB-R\fR option to force an unmount of any clone file systems
that are to be destroyed.
.RE

.RE

.sp
.ne 2
.na
\fB\fBzfs clone\fR [\fB-p\fR] [\fB-o\fR \fIproperty\fR=\fIvalue\fR] ...
\fIsnapshot\fR \fIfilesystem\fR|\fIvolume\fR\fR
.ad
.sp .6
.RS 4n
Creates a clone of the given snapshot. See the "Clones" section for details.
The target dataset can be located anywhere in the \fBZFS\fR hierarchy, and is
created as the same type as the original.
.sp
```

```
.ne 2
.na
\fB\fB-p\fR\fR
.ad
.sp .6
.RS 4n
Creates all the non-existing parent datasets. Datasets created in this manner
are automatically mounted according to the \fBmountpoint\fR property inherited
from their parent. If the target filesystem or volume already exists, the
operation completes successfully.
.RE

.sp
.ne 2
.na
\fB\fB-o\fR \fIproperty\fR=\fIvalue\fR\fR
.ad
.sp .6
.RS 4n
Sets the specified property; see \fBzfs create\fR for details.
.RE

.RE

.sp
.ne 2
.na
\fB\fBzfs promote\fR \fIclone-filesystem\fR\fR
.ad
.sp .6
.RS 4n
Promotes a clone file system to no longer be dependent on its "origin"
snapshot. This makes it possible to destroy the file system that the clone was
created from. The clone parent-child dependency relationship is reversed, so
that the origin file system becomes a clone of the specified file system.
.sp
The snapshot that was cloned, and any snapshots previous to this snapshot, are
now owned by the promoted clone. The space they use moves from the origin file
system to the promoted clone, so enough space must be available to accommodate
these snapshots. No new space is consumed by this operation, but the space
accounting is adjusted. The promoted clone must not have any conflicting
snapshot names of its own. The \fBrename\fR subcommand can be used to rename
any conflicting snapshots.
.RE

.sp
.ne 2
.na
\fB\fBzfs rename\fR [\fB-f\fR] \fIfilesystem\fR|\fIvolume\fR|\fIsnapshot\fR\fR
.ad
.br
.na
\fB\fIfilesystem\fR|\fIvolume\fR|\fIsnapshot\fR\fR
.ad
.br
.na
\fB\fBzfs rename\fR [\fB-fp\fR] \fIfilesystem\fR|\fIvolume\fR
\fIfilesystem\fR|\fIvolume\fR\fR
.ad
.sp .6
.RS 4n
Renames the given dataset. The new target can be located anywhere in the
\fBZFS\fR hierarchy, with the exception of snapshots. Snapshots can only be
renamed within the parent file system or volume. When renaming a snapshot, the
parent file system of the snapshot does not need to be specified as part of the
second argument. Renamed file systems can inherit new mount points, in which
```



```
case they are unmounted and remounted at the new mount point.
.sp
.ne 2
.na
\fB\fB-p\fR\fR
.ad
.sp .6
.RS 4n
Creates all the nonexistent parent datasets. Datasets created in this manner
are automatically mounted according to the \fBmountpoint\fR property inherited
from their parent.
.RE

.sp
.ne 2
.na
\fB\fB-f\fR\fR
.ad
.sp .6
.RS 4n
Force unmount any filesystems that need to be unmounted in the process.
.RE

.RE

.sp
.ne 2
.na
\fB\fBzfs rename\fR \fB-r\fR \fIsnapshot\fR \fIsnapshot\fR\fR
.ad
.sp .6
.RS 4n
Recursively rename the snapshots of all descendent datasets. Snapshots are the
only dataset that can be renamed recursively.
.RE

.sp
.ne 2
.na
\fB\fBzfs\fR \fBlist\fR [\fB-r\fR|\fB-d\fR \fIdepth\fR] [\fB-H\fR] [\fB-o\fR
\fIproperty\fR[,\fI\&...\fR]] [ \fB-t\fR \fItype\fR[,\fI\&...\fR]] [ \fB-s\fR
\fIproperty\fR ] ... [ \fB-S\fR \fIproperty\fR ] ...
[\fIfilesystem\fR|\fIvolume\fR|\fIsnapshot\fR] ...\fR
.ad
.sp .6
.RS 4n
Lists the property information for the given datasets in tabular form. If
specified, you can list property information by the absolute pathname or the
relative pathname. By default, all file systems and volumes are displayed.
Snapshots are displayed if the \fBlistsnaps\fR property is \fBon\fR (the
default is \fBoff\fR) . The following fields are displayed,
\fBname,used,available,referenced,mountpoint\fR.
.sp
.ne 2
.na
\fB\fB-H\fR\fR
.ad
.sp .6
.RS 4n
Used for scripting mode. Do not print headers and separate fields by a single
tab instead of arbitrary white space.
.RE

.sp
.ne 2
.na
```

```
2107 \fB\fB-r\fR\fR
2108 .ad
2109 .sp .6
2110 .RS 4n
2111 Recursively display any children of the dataset on the command line.
2112 .RE

2114 .sp
2115 .ne 2
2116 .na
2117 \fB\fB-d\fR \fIdepth\fR\fR
2118 .ad
2119 .sp .6
2120 .RS 4n
2121 Recursively display any children of the dataset, limiting the recursion to
2122 \fIdepth\fR. A depth of \fB1\fR will display only the dataset and its direct
2123 children.
2124 .RE

2126 .sp
2127 .ne 2
2128 .na
2129 \fB\fB-o\fR \fIproperty\fR\fR
2130 .ad
2131 .sp .6
2132 .RS 4n
2133 A comma-separated list of properties to display. The property must be:
2134 .RS +4
2135 .TP
2136 .ie t \(bu
2137 .el o
2138 One of the properties described in the "Native Properties" section
2139 .RE
2140 .RS +4
2141 .TP
2142 .ie t \(bu
2143 .el o
2144 A user property
2145 .RE
2146 .RS +4
2147 .TP
2148 .ie t \(bu
2149 .el o
2150 The value \fBname\fR to display the dataset name
2151 .RE
2152 .RS +4
2153 .TP
2154 .ie t \(bu
2155 .el o
2156 The value \fBspace\fR to display space usage properties on file systems and
2157 volumes. This is a shortcut for specifying \fB-o
2158 name,avail,used,usedsnap,usedds,usedrefreserv,usedchild\fR \fB-t
2159 filesystem,volume\fR syntax.
2160 .RE
2161 .RE

2163 .sp
2164 .ne 2
2165 .na
2166 \fB\fB-s\fR \fIproperty\fR\fR
2167 .ad
2168 .sp .6
2169 .RS 4n
2170 A property for sorting the output by column in ascending order based on the
2171 value of the property. The property must be one of the properties described in
2172 the "Properties" section, or the special value \fBname\fR to sort by the
```



```
2173 dataset name. Multiple properties can be specified at one time using multiple
2174 \fB-s\fR property options. Multiple \fB-s\fR options are evaluated from left to
2175 right in decreasing order of importance.
2176 .sp
2177 The following is a list of sorting criteria:
2178 .RS +4
2179 .TP
2180 .ie t \(bu
2181 .el o
2182 Numeric types sort in numeric order.
2183 .RE
2184 .RS +4
2185 .TP
2186 .ie t \(bu
2187 .el o
2188 String types sort in alphabetical order.
2189 .RE
2190 .RS +4
2191 .TP
2192 .ie t \(bu
2193 .el o
2194 Types inappropriate for a row sort that row to the literal bottom, regardless
2195 of the specified ordering.
2196 .RE
2197 .RS +4
2198 .TP
2199 .ie t \(bu
2200 .el o
2201 If no sorting options are specified the existing behavior of \fBzfs list\fR is
2202 preserved.
2203 .RE
2204 .RE

2206 .sp
2207 .ne 2
2208 .na
2209 \fB\fB-S\fR \fIproperty\fR\fR
2210 .ad
2211 .sp .6
2212 .RS 4n
2213 Same as the \fB-s\fR option, but sorts by property in descending order.
2214 .RE

2216 .sp
2217 .ne 2
2218 .na
2219 \fB\fB-t\fR \fItype\fR\fR
2220 .ad
2221 .sp .6
2222 .RS 4n
2223 A comma-separated list of types to display, where \fItype\fR is one of
2224 \fBfilesystem\fR, \fBsnapshot\fR , \fBvolume\fR, or \fBall\fR. For example,
2225 specifying \fB-t snapshot\fR displays only snapshots.
2226 .RE

2228 .RE

2230 .sp
2231 .ne 2
2232 .na
2233 \fB\fBzfs set\fR \fIproperty\fR=\fIvalue\fR
2234 \fIfilesystem\fR|\fIvolume\fR|\fIsnapshot\fR ...\fR
2235 .ad
2236 .sp .6
2237 .RS 4n
2238 Sets the property to the given value for each dataset. Only some properties can
```

```
2239 be edited. See the "Properties" section for more information on what properties
2240 can be set and acceptable values. Numeric values can be specified as exact
2241 values, or in a human-readable form with a suffix of \fBB\fR, \fBK\fR, \fBM\fR,
2242 \fBG\fR, \fBT\fR, \fBP\fR, \fBE\fR, \fBZ\fR (for bytes, kilobytes, megabytes,
2243 gigabytes, terabytes, petabytes, exabytes, or zettabytes, respectively). User
2244 properties can be set on snapshots. For more information, see the "User
2245 Properties" section.
2246 .RE

2248 .sp
2249 .ne 2
2250 .na
2251 \fB\fBzfs get\fR [\fB-r\fR|\fB-d\fR \fIdepth\fR] [\fB-Hp\fR] [\fB-o\fR
2252 \fIfield\fR[,...] [\fB-t\fR \fItype\fR[,...]] [\fB-s\fR \fIsource\fR[,...] "\fIa
2253 \fIproperty\fR[,...] \fIfilesystem\fR|\fIvolume\fR|\fIsnapshot\fR ...\fR
2254 .ad
2255 .sp .6
2256 .RS 4n
2257 Displays properties for the given datasets. If no datasets are specified, then
2258 the command displays properties for all datasets on the system. For each
2259 property, the following columns are displayed:
2260 .sp
2261 .in +2
2262 .nf
2263     name      Dataset name
2264      property  Property name
2265      value     Property value
2266      source    Property source. Can either be local, default,
2267                temporary, inherited, or none (-).
2268 .fi
2269 .in -2
2270 .sp

2272 All columns are displayed by default, though this can be controlled by using
2273 the \fB-o\fR option. This command takes a comma-separated list of properties as
2274 described in the "Native Properties" and "User Properties" sections.
2275 .sp
2276 The special value \fBall\fR can be used to display all properties that apply to
2277 the given dataset's type (filesystem, volume, or snapshot).
2278 .sp
2279 .ne 2
2280 .na
2281 \fB\fB-r\fR\fR
2282 .ad
2283 .sp .6
2284 .RS 4n
2285 Recursively display properties for any children.
2286 .RE

2288 .sp
2289 .ne 2
2290 .na
2291 \fB\fB-d\fR \fIdepth\fR\fR
2292 .ad
2293 .sp .6
2294 .RS 4n
2295 Recursively display any children of the dataset, limiting the recursion to
2296 \fIdepth\fR. A depth of \fB1\fR will display only the dataset and its direct
2297 children.
2298 .RE

2300 .sp
2301 .ne 2
2302 .na
2303 \fB\fB-H\fR\fR
2304 .ad
```



```
2305 .sp .6
2306 .RS 4n
2307 Display output in a form more easily parsed by scripts. Any headers are
2308 omitted, and fields are explicitly separated by a single tab instead of an
2309 arbitrary amount of space.
2310 .RE

2312 .sp
2313 .ne 2
2314 .na
2315 \fB\fB-o\fR \fIfield\fR\fR
2316 .ad
2317 .sp .6
2318 .RS 4n
2319 A comma-separated list of columns to display. \fBname,property,value,source\fR
2320 is the default value.
2321 .RE

2323 .sp
2324 .ne 2
2325 .na
2326 \fB\fB-s\fR \fIsource\fR\fR
2327 .ad
2328 .sp .6
2329 .RS 4n
2330 A comma-separated list of sources to display. Those properties coming from a
2331 source other than those in this list are ignored. Each source must be one of
2332 the following: \fBlocal,default,inherited,temporary,none\fR. The default value
2333 is all sources.
2334 .RE

2336 .sp
2337 .ne 2
2338 .na
2339 \fB\fB-p\fR\fR
2340 .ad
2341 .sp .6
2342 .RS 4n
2343 Display numbers in parseable (exact) values.
2344 .RE

2346 .RE

2348 .sp
2349 .ne 2
2350 .na
2351 \fB\fBzfs inherit\fR [\fB-r\fR] \fIproperty\fR
2352 \fIfilesystem\fR|\fIvolume\fR|\fIsnapshot\fR ...\fR
2353 .ad
2354 .sp .6
2355 .RS 4n
2356 Clears the specified property, causing it to be inherited from an ancestor. If
2357 no ancestor has the property set, then the default value is used. See the
2358 "Properties" section for a listing of default values, and details on which
2359 properties can be inherited.
2360 .sp
2361 .ne 2
2362 .na
2363 \fB\fB-r\fR\fR
2364 .ad
2365 .sp .6
2366 .RS 4n
2367 Recursively inherit the given property for all children.
2368 .RE

2370 .RE
```

```
.sp
.ne 2
.na
\fB\fBzfs upgrade\fR [\fB-v\fR]\fR
.ad
.sp .6
.RS 4n
Displays a list of file systems that are not the most recent version.
.RE

.sp
.ne 2
.na
\fB\fBzfs upgrade\fR [\fB-r\fR] [\fB-V\fR \fIversion\fR] [\fB-a\fR |
\fIfilesystem\fR]\fR
.ad
.sp .6
.RS 4n
Upgrades file systems to a new on-disk version. Once this is done, the file
systems will no longer be accessible on systems running older versions of the
software. \fBzfs send\fR streams generated from new snapshots of these file
systems cannot be accessed on systems running older versions of the software.
.sp
In general, the file system version is independent of the pool version. See
\fBzpool\fR(1M) for information on the \fBzpool upgrade\fR command.
.sp
In some cases, the file system version and the pool version are interrelated
and the pool version must be upgraded before the file system version can be
upgraded.
.sp
.ne 2
.na
\fB\fB-a\fR\fR
.ad
.sp .6
.RS 4n
Upgrade all file systems on all imported pools.
.RE

.sp
.ne 2
.na
\fB\fIfilesystem\fR\fR
.ad
.sp .6
.RS 4n
Upgrade the specified file system.
.RE

.sp
.ne 2
.na
\fB\fB-r\fR\fR
.ad
.sp .6
.RS 4n
Upgrade the specified file system and all descendent file systems
.RE

.sp
.ne 2
.na
\fB\fB-V\fR \fIversion\fR\fR
.ad
.sp .6
```



```
.RS 4n
Upgrade to the specified \fIversion\fR. If the \fB-V\fR flag is not specified,
this command upgrades to the most recent version. This option can only be used
to increase the version number, and only up to the most recent version
supported by this software.
.RE

.RE

.sp
.ne 2
.na
\fBzfs\fR \fBuserspace\fR [\fB-Hinp\fR] [\fB-o\fR \fIfield\fR[,...]]
[\fB-s\fR \fIfield\fR] ...
[\fB-S\fR \fIfield\fR] ...
[\fB-t\fR \fItype\fR[,...]] \fIfilesystem\fR|\fIsnapshot\fR
.ad
.sp .6
.RS 4n
Displays space consumed by, and quotas on, each user in the specified
filesystem or snapshot. This corresponds to the \fBuserused@\fR\fIuser\fR and
\fBuserquota@\fR\fIuser\fR properties.
.sp
.ne 2
.na
\fB\fB-n\fR\fR
.ad
.sp .6
.RS 4n
Print numeric ID instead of user/group name.
.RE

.sp
.ne 2
.na
\fB\fB-H\fR\fR
.ad
.sp .6
.RS 4n
Do not print headers, use tab-delimited output.
.RE

.sp
.ne 2
.na
\fB\fB-p\fR\fR
.ad
.sp .6
.RS 4n
Use exact (parsable) numeric output.
.RE

.sp
.ne 2
.na
\fB\fB-o\fR \fIfield\fR[,...]\fR
.ad
.sp .6
.RS 4n
Display only the specified fields from the following
set: \fBtype, name, used, quota\fR. The default is to display all fields.
.RE

.sp
.ne 2
.na
```

```
\fB\fB-s\fR \fIfield\fR\fR
.ad
.sp .6
.RS 4n
Sort output by this field. The \fIs\fR and \fIS\fR flags may be specified
multiple times to sort first by one field, then by another. The default is
\fB-s type\fR \fB-s name\fR.
.RE

.sp
.ne 2
.na
\fB\fB-S\fR \fIfield\fR\fR
.ad
.sp .6
.RS 4n
Sort by this field in reverse order. See \fB-s\fR.
.RE

.sp
.ne 2
.na
\fB\fB-t\fR \fItype\fR[,...]\fR
.ad
.sp .6
.RS 4n
Print only the specified types from the following
set: \fBall, posixuser, smbuser, posixgroup, smbgroup\fR. The default
is \fB-t posixuser,smbuser\fR. The default can be changed to include group
types.
.RE

.sp
.ne 2
.na
\fB\fB-i\fR\fR
.ad
.sp .6
.RS 4n
Translate SID to POSIX ID. The POSIX ID may be ephemeral if no mapping exists.
Normal POSIX interfaces (for example, \fBstat\fR(2), \fBls\fR \fB-l\fR) perform
this translation, so the \fB-i\fR option allows the output from \fBzfs
userspace\fR to be compared directly with those utilities. However, \fB-i\fR
may lead to confusion if some files were created by an SMB user before a
SMB-to-POSIX name mapping was established. In such a case, some files will be ow
by the SMB entity and some by the POSIX entity. However, the \fB-i\fR option
will report that the POSIX entity has the total usage and quota for both.
.RE

.RE

.sp
.ne 2
.na
\fBzfs\fR \fBgroupspace\fR [\fB-Hinp\fR] [\fB-o\fR \fIfield\fR[,...]]
[\fB-s\fR \fIfield\fR] ...
[\fB-S\fR \fIfield\fR] ...
[\fB-t\fR \fItype\fR[,...]] \fIfilesystem\fR|\fIsnapshot\fR
.ad
.sp .6
.RS 4n
Displays space consumed by, and quotas on, each group in the specified
filesystem or snapshot. This subcommand is identical to \fBzfs userspace\fR,
except that the default types to display are \fB-t posixgroup,smbgroup\fR.
.RE
```



```
.sp
.ne 2
.na
\fB\fBzfs mount\fR\fR
.ad
.sp .6
.RS 4n
Displays all \fBZFS\fR file systems currently mounted.
.RE

.sp
.ne 2
.na
\fB\fBzfs mount\fR [\fB-vO\fR] [\fB-o\fR \fIoptions\fR] \fB-a\fR |
\fIfilesystem\fR\fR
.ad
.sp .6
.RS 4n
Mounts \fBZFS\fR file systems. Invoked automatically as part of the boot
process.
.sp
.ne 2
.na
\fB\fB-o\fR \fIoptions\fR\fR
.ad
.sp .6
.RS 4n
An optional, comma-separated list of mount options to use temporarily for the
duration of the mount. See the "Temporary Mount Point Properties" section for
details.
.RE

.sp
.ne 2
.na
\fB\fB-O\fR\fR
.ad
.sp .6
.RS 4n
Perform an overlay mount. See \fBmount\fR(1M) for more information.
.RE

.sp
.ne 2
.na
\fB\fB-v\fR\fR
.ad
.sp .6
.RS 4n
Report mount progress.
.RE

.sp
.ne 2
.na
\fB\fB-a\fR\fR
.ad
.sp .6
.RS 4n
Mount all available \fBZFS\fR file systems. Invoked automatically as part of
the boot process.
.RE

.sp
.ne 2
.na
```

```
2635 \fB\fIfilesystem\fR\fR
2636 .ad
2637 .sp .6
2638 .RS 4n
2639 Mount the specified filesystem.
2640 .RE

2642 .RE

2644 .sp
2645 .ne 2
2646 .na
2647 \fB\fBzfs unmount\fR [\fB-f\fR] \fB-a\fR | \fIfilesystem\fR|\fImountpoint\fR\fR
2648 .ad
2649 .sp .6
2650 .RS 4n
2651 Unmounts currently mounted \fBZFS\fR file systems. Invoked automatically as
2652 part of the shutdown process.
2653 .sp
2654 .ne 2
2655 .na
2656 \fB\fB-f\fR\fR
2657 .ad
2658 .sp .6
2659 .RS 4n
2660 Forcefully unmount the file system, even if it is currently in use.
2661 .RE

2663 .sp
2664 .ne 2
2665 .na
2666 \fB\fB-a\fR\fR
2667 .ad
2668 .sp .6
2669 .RS 4n
2670 Unmount all available \fBZFS\fR file systems. Invoked automatically as part of
2671 the boot process.
2672 .RE

2674 .sp
2675 .ne 2
2676 .na
2677 \fB\fIfilesystem\fR|\fImountpoint\fR\fR
2678 .ad
2679 .sp .6
2680 .RS 4n
2681 Unmount the specified filesystem. The command can also be given a path to a
2682 \fBZFS\fR file system mount point on the system.
2683 .RE

2685 .RE

2687 .sp
2688 .ne 2
2689 .na
2690 \fB\fBzfs share\fR \fB-a\fR | \fIfilesystem\fR\fR
2691 .ad
2692 .sp .6
2693 .RS 4n
2694 Shares available \fBZFS\fR file systems.
2695 .sp
2696 .ne 2
2697 .na
2698 \fB\fB-a\fR\fR
2699 .ad
2700 .sp .6
```



```
2701 .RS 4n
2702 Share all available \fBZFS\fR file systems. Invoked automatically as part of
2703 the boot process.
2704 .RE

2706 .sp
2707 .ne 2
2708 .na
2709 \fB\fIfilesystem\fR\fR
2710 .ad
2711 .sp .6
2712 .RS 4n
2713 Share the specified filesystem according to the \fBsharenfs\fR and
2714 \fBsharesmb\fR properties. File systems are shared when the \fBsharenfs\fR or
2715 \fBsharesmb\fR property is set.
2716 .RE

2718 .RE

2720 .sp
2721 .ne 2
2722 .na
2723 \fB\fBzfs unshare\fR \fB-a\fR | \fIfilesystem\fR|\fImountpoint\fR\fR
2724 .ad
2725 .sp .6
2726 .RS 4n
2727 Unshares currently shared \fBZFS\fR file systems. This is invoked automatically
2728 as part of the shutdown process.
2729 .sp
2730 .ne 2
2731 .na
2732 \fB\fB-a\fR\fR
2733 .ad
2734 .sp .6
2735 .RS 4n
2736 Unshare all available \fBZFS\fR file systems. Invoked automatically as part of
2737 the boot process.
2738 .RE

2740 .sp
2741 .ne 2
2742 .na
2743 \fB\fIfilesystem\fR|\fImountpoint\fR\fR
2744 .ad
2745 .sp .6
2746 .RS 4n
2747 Unshare the specified filesystem. The command can also be given a path to a
2748 \fBZFS\fR file system shared on the system.
2749 .RE

2751 .RE

2753 .sp
2754 .ne 2
2755 .na
2756 \fBzfs send\fR [\fB-DnPpRv\fR] [\fB-\fR[\fBiI\fR] \fIsnapshot\fR] \fIsnapshot\fR
2757 .ad
2758 .sp .6
2759 .RS 4n
2760 Creates a stream representation of the second \fIsnapshot\fR, which is written
2761 to standard output. The output can be redirected to a file or to a different
2762 system (for example, using \fBssh\fR(1). By default, a full stream is
2763 generated.
2764 .sp
2765 .ne 2
2766 .na
```

\fB\fB-i\fR \fIsnapshot\fR\fR
.ad
.sp .6
.RS 4n
Generate an incremental stream from the first \fIsnapshot\fR to the second \fIsnapshot\fR. The incremental source (the first \fIsnapshot\fR) can be specified as the last component of the snapshot name (for example, the part after the \fB@\fR), and it is assumed to be from the same file system as the second \fIsnapshot\fR.
.sp
If the destination is a clone, the source may be the origin snapshot, which must be fully specified (for example, \fBpool/fs@origin\fR, not just \fB@origin\fR).
.RE

.sp
.ne 2
.na
\fB\fB-I\fR \fIsnapshot\fR\fR
.ad
.sp .6
.RS 4n
Generate a stream package that sends all intermediary snapshots from the first snapshot to the second snapshot. For example, \fB-I @a fs@d\fR is similar to \fB-i @a fs@b; -i @b fs@c; -i @c fs@d\fR. The incremental source snapshot may be specified as with the \fB-i\fR option.
.RE

.sp
.ne 2
.na
\fB\fB-R\fR\fR
.ad
.sp .6
.RS 4n
Generate a replication stream package, which will replicate the specified filesystem, and all descendent file systems, up to the named snapshot. When received, all properties, snapshots, descendent file systems, and clones are preserved.
.sp
If the \fB-i\fR or \fB-I\fR flags are used in conjunction with the \fB-R\fR flag, an incremental replication stream is generated. The current values of properties, and current snapshot and file system names are set when the stream is received. If the \fB-F\fR flag is specified when this stream is received, snapshots and file systems that do not exist on the sending side are destroyed.
.RE

.sp
.ne 2
.na
\fB\fB-D\fR\fR
.ad
.sp .6
.RS 4n
Generate a deduplicated stream. Blocks which would have been sent multiple times in the send stream will only be sent once. The receiving system must also support this feature to recieve a deduplicated stream. This flag can be used regardless of the dataset's \fBdedup\fR property, but performance will be much better if the filesystem uses a dedup-capable checksum (eg. \fBsha256\fR).
.RE

.sp
.ne 2
.na
\fB\fB-p\fR\fR

.ad
.sp .6
.RS 4n
Include the dataset's properties in the stream. This flag is implicit when \fB-R\fR is specified. The receiving system must also support this feature.
.RE

.sp
.ne 2
.na
\fB\fB-n\fR\fR
.ad
.sp .6
.RS 4n
Do a dry-run ("No-op") send. Do not generate any actual send data. This is useful in conjunction with the \fB-v\fR or \fB-P\fR flags to determine what data will be sent.
.RE

.sp
.ne 2
.na
\fB\fB-P\fR\fR
.ad
.sp .6
.RS 4n
Print machine-parsable verbose information about the stream package generated.
.RE

.sp
.ne 2
.na
\fB\fB-v\fR\fR
.ad
.sp .6
.RS 4n
Print verbose information about the stream package generated. This information includes a per-second report of how much data has been sent.
.RE

The format of the stream is committed. You will be able to receive your streams on future versions of \fBZFS\fR.
.RE

.sp
.ne 2
.na
\fB\fBzfs receive\fR [\fB-vnFu\fR]
\fIfilesystem\fR|\fIvolume\fR|\fIsnapshot\fR\fR
.ad
.br
.na
\fB\fBzfs receive\fR [\fB-vnFu\fR] [\fB-d\fR|\fB-e\fR] \fIfilesystem\fR\fR
.ad
.sp .6
.RS 4n
Creates a snapshot whose contents are as specified in the stream provided on standard input. If a full stream is received, then a new file system is created as well. Streams are created using the \fBzfs send\fR subcommand, which by default creates a full stream. \fBzfs recv\fR can be used as an alias for \fBzfs receive\fR.
.sp
If an incremental stream is received, then the destination file system must already exist, and its most recent snapshot must match the incremental stream's source. For \fBzvols\fR, the destination device link is destroyed and recreated, which means the \fBzvol\fR cannot be accessed during the

\fBreceive\fR operation.
.sp
When a snapshot replication package stream that is generated by using the \fBzfs send\fR \fB-R\fR command is received, any snapshots that do not exist on the sending location are destroyed by using the \fBzfs destroy\fR \fB-d\fR command.
.sp
The name of the snapshot (and file system, if a full stream is received) that this subcommand creates depends on the argument type and the use of the \fB-d\fR or \fB-e\fR options.
.sp
If the argument is a snapshot name, the specified \fIsnapshot\fR is created. If the argument is a file system or volume name, a snapshot with the same name as the sent snapshot is created within the specified \fIfilesystem\fR or \fIvolume\fR. If neither of the \fB-d\fR or \fB-e\fR options are specified, the provided target snapshot name is used exactly as provided.
.sp
The \fB-d\fR and \fB-e\fR options cause the file system name of the target snapshot to be determined by appending a portion of the sent snapshot’s name to the specified target \fIfilesystem\fR. If the \fB-d\fR option is specified, all but the first element of the sent snapshot’s file system path (usually the pool name) is used and any required intermediate file systems within the specified one are created. If the \fB-e\fR option is specified, then only the last element of the sent snapshot’s file system name (i.e. the name of the source file system itself) is used as the target file system name.
.sp
.ne 2
.na
\fB\fB-d\fR\fR
.ad
.sp .6
.RS 4n
Discard the first element of the sent snapshot’s file system name, using the remaining elements to determine the name of the target file system for the new snapshot as described in the paragraph above.
.RE

.sp
.ne 2
.na
\fB\fB-e\fR\fR
.ad
.sp .6
.RS 4n
Discard all but the last element of the sent snapshot’s file system name, using that element to determine the name of the target file system for the new snapshot as described in the paragraph above.
.RE

.sp
.ne 2
.na
\fB\fB-u\fR\fR
.ad
.sp .6
.RS 4n
File system that is associated with the received stream is not mounted.
.RE

.sp
.ne 2
.na
\fB\fB-v\fR\fR
.ad
.sp .6
.RS 4n

Print verbose information about the stream and the time required to perform the receive operation.
.RE

.sp
.ne 2
.na
\fB\fB-n\fR\fR
.ad
.sp .6
.RS 4n
Do not actually receive the stream. This can be useful in conjunction with the \fB-v\fR option to verify the name the receive operation would use.
.RE

.sp
.ne 2
.na
\fB\fB-F\fR\fR
.ad
.sp .6
.RS 4n
Force a rollback of the file system to the most recent snapshot before performing the receive operation. If receiving an incremental replication stream (for example, one generated by \fBzfs send -R -[iI]\fR), destroy snapshots and file systems that do not exist on the sending side.
.RE

.RE

.sp
.ne 2
.na
\fB\fBzfs allow\fR \fIfilesystem\fR | \fIvolume\fR\fR
.ad
.sp .6
.RS 4n
Displays permissions that have been delegated on the specified filesystem or volume. See the other forms of \fBzfs allow\fR for more information.
.RE

.sp
.ne 2
.na
\fB\fBzfs allow\fR [\fB-ldug\fR] "\fIeveryone\fR"|\fIuser\fR|\fIgroup\fR[,...] \fIperm\fR|@\fIsetname\fR[,...] \fIfilesystem\fR| \fIvolume\fR\fR
.ad
.br
.na
\fB\fBzfs allow\fR [\fB-ld\fR] \fB-e\fR \fIperm\fR|@\fIsetname\fR[,...] \fIfilesystem\fR | \fIvolume\fR\fR
.ad
.sp .6
.RS 4n
Delegates \fBZFS\fR administration permission for the file systems to non-privileged users.
.sp
.ne 2
.na
\fB[\fB-ug\fR] "\fIeveryone\fR"|\fIuser\fR|\fIgroup\fR[,...]\fR
.ad
.sp .6
.RS 4n
Specifies to whom the permissions are delegated. Multiple entities can be specified as a comma-separated list. If neither of the \fB-ug\fR options are specified, then the argument is interpreted preferentially as the keyword

"everyone", then as a user name, and lastly as a group name. To specify a user or group named "everyone", use the \fB-u\fR or \fB-g\fR options. To specify a group with the same name as a user, use the \fB-g\fR options.
.RE

.sp
.ne 2
.na
\fB[\fB-e\fR] \fIperm\fR|@\fIsetname\fR[,...]\fR
.ad
.sp .6
.RS 4n
Specifies that the permissions be delegated to "everyone." Multiple permissions may be specified as a comma-separated list. Permission names are the same as \fBZFS\fR subcommand and property names. See the property list below. Property set names, which begin with an at sign (\fB@\fR) , may be specified. See the \fB-s\fR form below for details.
.RE

.sp
.ne 2
.na
\fB[\fB-ld\fR] \fIfilesystem\fR|\fIvolume\fR\fR
.ad
.sp .6
.RS 4n
Specifies where the permissions are delegated. If neither of the \fB-ld\fR options are specified, or both are, then the permissions are allowed for the file system or volume, and all of its descendents. If only the \fB-l\fR option is used, then is allowed "locally" only for the specified file system. If only the \fB-d\fR option is used, then is allowed only for the descendent file systems.
.RE

.RE

.sp
.LP
Permissions are generally the ability to use a \fBZFS\fR subcommand or change a \fBZFS\fR property. The following permissions are available:
.sp
.in +2
.nf
NAME   TYPE   NOTES
allow   subcommand   Must also have the permission that is being
                              allowed
clone   subcommand   Must also have the 'create' ability and 'mount'
                              ability in the origin file system
create   subcommand   Must also have the 'mount' ability
destroy   subcommand   Must also have the 'mount' ability
diff   subcommand   Allows lookup of paths within a dataset
                              given an object number, and the ability to
                              create snapshots necessary to 'zfs diff'.
mount   subcommand   Allows mount/umount of ZFS datasets
promote   subcommand   Must also have the 'mount'
                              and 'promote' ability in the origin file system
receive   subcommand   Must also have the 'mount' and 'create' ability
rename   subcommand   Must also have the 'mount' and 'create'
                              ability in the new parent
rollback   subcommand   Must also have the 'mount' ability
send   subcommand
share   subcommand   Allows sharing file systems over NFS or SMB
                              protocols
snapshot   subcommand   Must also have the 'mount' ability
groupquota   other   Allows accessing any groupquota@... property
groupused   other   Allows reading any groupused@... property

userprop   other   Allows changing any user property
userquota   other   Allows accessing any userquota@... property
userused   other   Allows reading any userused@... property

aclinherit   property
aclmode   property
atime   property
canmount   property
casesensitivity property
checksum   property
compression   property
copies   property
devices   property
exec   property
mountpoint   property
nbmand   property
normalization   property
primarycache   property
quota   property
readonly   property
recordsize   property
refquota   property
refreservation property
reservation   property
secondarycache property
setuid   property
shareiscsi   property
sharenfs   property
sharesmb   property
snapdir   property
utf8only   property
version   property
volblocksize   property
volsize   property
vscan   property
xattr   property
zoned   property
.fi
.in -2
.sp

.sp
.ne 2
.na
\fB\fBzfs allow\fR \fB-c\fR \fIperm\fR|@\fIsetname\fR[,...]
\fIfilesystem\fR|\fIvolume\fR\fR
.ad
.sp .6
.RS 4n
Sets "create time" permissions. These permissions are granted (locally) to the creator of any newly-created descendent file system.
.RE

.sp
.ne 2
.na
\fB\fBzfs allow\fR \fB-s\fR @\fIsetname\fR \fIperm\fR|@\fIsetname\fR[,...]
\fIfilesystem\fR|\fIvolume\fR\fR
.ad
.sp .6
.RS 4n
Defines or adds permissions to a permission set. The set can be used by other \fBzfs allow\fR commands for the specified file system and its descendents. Sets are evaluated dynamically, so changes to a set are immediately reflected. Permission sets follow the same naming restrictions as ZFS file systems, but the name must begin with an "at sign" (\fB@\fR), and can be no more than 64

```
3163 characters long.
3164 .RE

3166 .sp
3167 .ne 2
3168 .na
3169 \fB\fBzfs unallow\fR [\fB-rldug\fR]
3170 "\fIeveryone\fR"|\fIuser\fR|\fIgroup\fR[,...]
3171 [\fIperm\fR|@\fIsetname\fR[, ...]] \fIfilesystem\fR|\fIvolume\fR\fR
3172 .ad
3173 .br
3174 .na
3175 \fB\fBzfs unallow\fR [\fB-rld\fR] \fB-e\fR [\fIperm\fR|@\fIsetname\fR [,...]]
3176 \fIfilesystem\fR|\fIvolume\fR\fR
3177 .ad
3178 .br
3179 .na
3180 \fB\fBzfs unallow\fR [\fB-r\fR] \fB-c\fR [\fIperm\fR|@\fIsetname\fR[,...]]\fR
3181 .ad
3182 .br
3183 .na
3184 \fB\fIfilesystem\fR|\fIvolume\fR\fR
3185 .ad
3186 .sp .6
3187 .RS 4n
3188 Removes permissions that were granted with the \fBzfs allow\fR command. No
3189 permissions are explicitly denied, so other permissions granted are still in
3190 effect. For example, if the permission is granted by an ancestor. If no
3191 permissions are specified, then all permissions for the specified \fIuser\fR,
3192 \fIgroup\fR, or \fIeveryone\fR are removed. Specifying "everyone" (or using the
3193 \fB-e\fR option) only removes the permissions that were granted to "everyone",
3194 not all permissions for every user and group. See the \fBzfs allow\fR command
3195 for a description of the \fB-ldugec\fR options.
3196 .sp
3197 .ne 2
3198 .na
3199 \fB\fB-r\fR\fR
3200 .ad
3201 .sp .6
3202 .RS 4n
3203 Recursively remove the permissions from this file system and all descendents.
3204 .RE

3206 .RE

3208 .sp
3209 .ne 2
3210 .na
3211 \fB\fBzfs unallow\fR [\fB-r\fR] \fB-s\fR @\fIsetname\fR
3212 [\fIperm\fR|@\fIsetname\fR[,...]]\fR
3213 .ad
3214 .br
3215 .na
3216 \fB\fIfilesystem\fR|\fIvolume\fR\fR
3217 .ad
3218 .sp .6
3219 .RS 4n
3220 Removes permissions from a permission set. If no permissions are specified,
3221 then all permissions are removed, thus removing the set entirely.
3222 .RE

3224 .sp
3225 .ne 2
3226 .na
3227 \fB\fBzfs hold\fR [\fB-r\fR] \fItag\fR \fIsnapshot\fR...\fR
3228 .ad
```



```
3229 .sp .6
3230 .RS 4n
3231 Adds a single reference, named with the \fItag\fR argument, to the specified
3232 snapshot or snapshots. Each snapshot has its own tag namespace, and tags must
3233 be unique within that space.
3234 .sp
3235 If a hold exists on a snapshot, attempts to destroy that snapshot by using the
3236 \fBzfs destroy\fR command return \fBEBUSY\fR.
3237 .sp
3238 .ne 2
3239 .na
3240 \fB\fB-r\fR\fR
3241 .ad
3242 .sp .6
3243 .RS 4n
3244 Specifies that a hold with the given tag is applied recursively to the
3245 snapshots of all descendent file systems.
3246 .RE

3248 .RE

3250 .sp
3251 .ne 2
3252 .na
3253 \fB\fBzfs holds\fR [\fB-r\fR] \fIsnapshot\fR...\fR
3254 .ad
3255 .sp .6
3256 .RS 4n
3257 Lists all existing user references for the given snapshot or snapshots.
3258 .sp
3259 .ne 2
3260 .na
3261 \fB\fB-r\fR\fR
3262 .ad
3263 .sp .6
3264 .RS 4n
3265 Lists the holds that are set on the named descendent snapshots, in addition to
3266 listing the holds on the named snapshot.
3267 .RE

3269 .RE

3271 .sp
3272 .ne 2
3273 .na
3274 \fB\fBzfs release\fR [\fB-r\fR] \fItag\fR \fIsnapshot\fR...\fR
3275 .ad
3276 .sp .6
3277 .RS 4n
3278 Removes a single reference, named with the \fItag\fR argument, from the
3279 specified snapshot or snapshots. The tag must already exist for each snapshot.
3280 .sp
3281 If a hold exists on a snapshot, attempts to destroy that snapshot by using the
3282 \fBzfs destroy\fR command return \fBEBUSY\fR.
3283 .sp
3284 .ne 2
3285 .na
3286 \fB\fB-r\fR\fR
3287 .ad
3288 .sp .6
3289 .RS 4n
3290 Recursively releases a hold with the given tag on the snapshots of all
3291 descendent file systems.
3292 .RE

3294 .sp
```

```
.ne 2
.na
\fBzfs diff\fR [\fB-FHt\fR] \fIsnapshot\fR \fIsnapshot|filesystem\fR
.ad
.sp .6
.RS 4n
Display the difference between a snapshot of a given filesystem and another
snapshot of that filesystem from a later time or the current contents of the
filesystem.  The first column is a character indicating the type of change,
the other columns indicate pathname, new pathname (in case of rename), change
in link count, and optionally file type and/or change time.

The types of change are:
.in +2
.nf
-       The path has been removed
+       The path has been created
M       The path has been modified
R       The path has been renamed
.fi
.in -2
.sp
.ne 2
.na
\fB-F\fR
.ad
.sp .6
.RS 4n
Display an indication of the type of file, in a manner similar to the \fB-F\fR
option of \fBls\fR(1).
.in +2
.nf
B       Block device
C       Character device
/       Directory
>       Door
|       Named pipe
@       Symbolic link
P       Event port
=       Socket
F       Regular file
.fi
.in -2
.RE
.sp
.ne 2
.na
\fB-H\fR
.ad
.sp .6
.RS 4n
Give more parseable tab-separated output, without header lines and without arrow
.RE
.sp
.ne 2
.na
\fB-t\fR
.ad
.sp .6
.RS 4n
Display the path's inode change time as the first column of output.
.RE

.SH EXAMPLES
.LP
\fBExample 1 \fRCreating a ZFS File System Hierarchy
```



```
.sp
.LP
The following commands create a file system named \fBpool/home\fR and a file
system named \fBpool/home/bob\fR. The mount point \fB/export/home\fR is set for
the parent file system, and is automatically inherited by the child file
system.

.sp
.in +2
.nf
# \fBzfs create pool/home\fR
# \fBzfs set mountpoint=/export/home pool/home\fR
# \fBzfs create pool/home/bob\fR
.fi
.in -2
.sp

.LP
\fBExample 2 \fRCreating a ZFS Snapshot
.sp
.LP
The following command creates a snapshot named \fByesterday\fR. This snapshot
is mounted on demand in the \fB\&.zfs/snapshot\fR directory at the root of the
\fBpool/home/bob\fR file system.

.sp
.in +2
.nf
# \fBzfs snapshot pool/home/bob@yesterday\fR
.fi
.in -2
.sp

.LP
\fBExample 3 \fRCreating and Destroying Multiple Snapshots
.sp
.LP
The following command creates snapshots named \fByesterday\fR of
\fBpool/home\fR and all of its descendent file systems. Each snapshot is
mounted on demand in the \fB\&.zfs/snapshot\fR directory at the root of its
file system. The second command destroys the newly created snapshots.

.sp
.in +2
.nf
# \fBzfs snapshot -r pool/home@yesterday\fR
# \fBzfs destroy -r pool/home@yesterday\fR
.fi
.in -2
.sp

.LP
\fBExample 4 \fRDisabling and Enabling File System Compression
.sp
.LP
The following command disables the \fBcompression\fR property for all file
systems under \fBpool/home\fR. The next command explicitly enables
\fBcompression\fR for \fBpool/home/anne\fR.

.sp
.in +2
.nf
# \fBzfs set compression=off pool/home\fR
# \fBzfs set compression=on pool/home/anne\fR
.fi
.in -2
```

```
.sp

.LP
\fBExample 5 \fRListing ZFS Datasets
.sp
.LP
The following command lists all active file systems and volumes in the system.
Snapshots are displayed if the \fBlistsnaps\fR property is \fBon\fR. The
default is \fBoff\fR. See \fBzpool\fR(1M) for more information on pool
properties.

.sp
.in +2
.nf
# \fBzfs list\fR
   NAME                    USED  AVAIL  REFER  MOUNTPOINT
   pool                    450K   457G    18K  /pool
   pool/home               315K   457G    21K  /export/home
   pool/home/anne           18K   457G    18K  /export/home/anne
   pool/home/bob           276K   457G   276K  /export/home/bob
.fi
.in -2
.sp

.LP
\fBExample 6 \fRSetting a Quota on a ZFS File System
.sp
.LP
The following command sets a quota of 50 Gbytes for \fBpool/home/bob\fR.

.sp
.in +2
.nf
# \fBzfs set quota=50G pool/home/bob\fR
.fi
.in -2
.sp

.LP
\fBExample 7 \fRListing ZFS Properties
.sp
.LP
The following command lists all properties for \fBpool/home/bob\fR.

.sp
.in +2
.nf
# \fBzfs get all pool/home/bob\fR
NAME           PROPERTY              VALUE                  SOURCE
pool/home/bob  type                  filesystem             -
pool/home/bob  creation              Tue Jul 21 15:53 2009  -
pool/home/bob  used                  21K                    -
pool/home/bob  available             20.0G                  -
pool/home/bob  referenced            21K                    -
pool/home/bob  compressratio         1.00x                  -
pool/home/bob  mounted               yes                    -
pool/home/bob  quota                 20G                    local
pool/home/bob  reservation           none                   default
pool/home/bob  recordsize            128K                   default
pool/home/bob  mountpoint            /pool/home/bob         default
pool/home/bob  sharenfs              off                    default
pool/home/bob  checksum              on                     default
pool/home/bob  compression           on                     local
pool/home/bob  atime                 on                     default
pool/home/bob  devices               on                     default
pool/home/bob  exec                  on                     default
```



```
pool/home/bob  setuid                on                     default
pool/home/bob  readonly              off                    default
pool/home/bob  zoned                 off                    default
pool/home/bob  snapdir               hidden                 default
pool/home/bob  aclmode               discard                default
pool/home/bob  aclinherit            restricted             default
pool/home/bob  canmount              on                     default
pool/home/bob  shareiscsi            off                    default
pool/home/bob  xattr                 on                     default
pool/home/bob  copies                1                      default
pool/home/bob  version               4                      -
pool/home/bob  utf8only              off                    -
pool/home/bob  normalization         none                   -
pool/home/bob  casesensitivity       sensitive              -
pool/home/bob  vscan                 off                    default
pool/home/bob  nbmand                off                    default
pool/home/bob  sharesmb              off                    default
pool/home/bob  refquota              none                   default
pool/home/bob  refreservation        none                   default
pool/home/bob  primarycache          all                    default
pool/home/bob  secondarycache        all                    default
pool/home/bob  usedbysnapshots       0                      -
pool/home/bob  usedbydataset         21K                    -
pool/home/bob  usedbychildren        0                      -
pool/home/bob  usedbyrefreservation  0                      -
.fi
.in -2
.sp

.sp
.LP
The following command gets a single property value.

.sp
.in +2
.nf
# \fBzfs get -H -o value compression pool/home/bob\fR
on
.fi
.in -2
.sp

.sp
.LP
The following command lists all properties with local settings for
\fBpool/home/bob\fR.

.sp
.in +2
.nf
# \fBzfs get -r -s local -o name,property,value all pool/home/bob\fR
NAME           PROPERTY              VALUE
pool/home/bob  quota                 20G
pool/home/bob  compression           on
.fi
.in -2
.sp

.LP
\fBExample 8 \fRRolling Back a ZFS File System
.sp
.LP
The following command reverts the contents of \fBpool/home/anne\fR to the
snapshot named \fByesterday\fR, deleting all intermediate snapshots.

.sp
```

```
.in +2
.nf
# \fBzfs rollback -r pool/home/anne@yesterday\fR
.fi
.in -2
.sp

.LP
\fBExample 9 \fRCreating a ZFS Clone
.sp
.LP
The following command creates a writable file system whose initial contents are
the same as \fBpool/home/bob@yesterday\fR.

.sp
.in +2
.nf
# \fBzfs clone pool/home/bob@yesterday pool/clone\fR
.fi
.in -2
.sp

.LP
\fBExample 10 \fRPromoting a ZFS Clone
.sp
.LP
The following commands illustrate how to test out changes to a file system, and
then replace the original file system with the changed one, using clones, clone
promotion, and renaming:

.sp
.in +2
.nf
# \fBzfs create pool/project/production\fR
  populate /pool/project/production with data
# \fBzfs snapshot pool/project/production@today\fR
# \fBzfs clone pool/project/production@today pool/project/beta\fR
make changes to /pool/project/beta and test them
# \fBzfs promote pool/project/beta\fR
# \fBzfs rename pool/project/production pool/project/legacy\fR
# \fBzfs rename pool/project/beta pool/project/production\fR
once the legacy version is no longer needed, it can be destroyed
# \fBzfs destroy pool/project/legacy\fR
.fi
.in -2
.sp

.LP
\fBExample 11 \fRInheriting ZFS Properties
.sp
.LP
The following command causes \fBpool/home/bob\fR and \fBpool/home/anne\fR to
inherit the \fBchecksum\fR property from their parent.

.sp
.in +2
.nf
# \fBzfs inherit checksum pool/home/bob pool/home/anne\fR
.fi
.in -2
.sp

.LP
\fBExample 12 \fRRemotely Replicating ZFS Data
.sp
.LP
```



```
The following commands send a full stream and then an incremental stream to a
remote machine, restoring them into \fBpoolB/received/fs@a\fRand
\fBpoolB/received/fs@b\fR, respectively. \fBpoolB\fR must contain the file
system \fBpoolB/received\fR, and must not initially contain
\fBpoolB/received/fs\fR.

.sp
.in +2
.nf
# \fBzfs send pool/fs@a | \e\fR
    \fBssh host zfs receive poolB/received/fs@a\fR
# \fBzfs send -i a pool/fs@b | ssh host \e\fR
    \fBzfs receive poolB/received/fs\fR
.fi
.in -2
.sp

.LP
\fBExample 13 \fRUsing the \fBzfs receive\fR \fB-d\fR Option
.sp
.LP
The following command sends a full stream of \fBpoolA/fsA/fsB@snap\fR to a
remote machine, receiving it into \fBpoolB/received/fsA/fsB@snap\fR. The
\fBfsA/fsB@snap\fR portion of the received snapshot's name is determined from
the name of the sent snapshot. \fBpoolB\fR must contain the file system
\fBpoolB/received\fR. If \fBpoolB/received/fsA\fR does not exist, it is created
as an empty file system.

.sp
.in +2
.nf
# \fBzfs send poolA/fsA/fsB@snap | \e
    ssh host zfs receive -d poolB/received\fR
.fi
.in -2
.sp

.LP
\fBExample 14 \fRSetting User Properties
.sp
.LP
The following example sets the user-defined \fBcom.example:department\fR
property for a dataset.

.sp
.in +2
.nf
# \fBzfs set com.example:department=12345 tank/accounting\fR
.fi
.in -2
.sp

.LP
\fBExample 15 \fRCreating a ZFS Volume as an iSCSI Target Device
.sp
.LP
The following example shows how to create a \fBZFS\fR volume as an \fBiSCSI\fR
target.

.sp
.in +2
.nf
# \fBzfs create -V 2g pool/volumes/vol1\fR
# \fBzfs set shareiscsi=on pool/volumes/vol1\fR
# \fBiscsitadm list target\fR
Target: pool/volumes/vol1
```

```
 iSCSI Name:
 iqn.1986-03.com.sun:02:7b4b02a6-3277-eb1b-e686-a24762c52a8c
 Connections: 0
.fi
.in -2
.sp

.sp
.LP
After the \fBiSCSI\fR target is created, set up the \fBiSCSI\fR initiator. For
more information about the Solaris \fBiSCSI\fR initiator, see
\fBiscsitadm\fR(1M).
.LP
\fBExample 16 \fRPerforming a Rolling Snapshot
.sp
.LP
The following example shows how to maintain a history of snapshots with a
consistent naming scheme. To keep a week's worth of snapshots, the user
destroys the oldest snapshot, renames the remaining snapshots, and then creates
a new snapshot, as follows:

.sp
.in +2
.nf
# \fBzfs destroy -r pool/users@7daysago\fR
# \fBzfs rename -r pool/users@6daysago @7daysago\fR
# \fBzfs rename -r pool/users@5daysago @6daysago\fR
# \fBzfs rename -r pool/users@yesterday @5daysago\fR
# \fBzfs rename -r pool/users@yesterday @4daysago\fR
# \fBzfs rename -r pool/users@yesterday @3daysago\fR
# \fBzfs rename -r pool/users@yesterday @2daysago\fR
# \fBzfs rename -r pool/users@today @yesterday\fR
# \fBzfs snapshot -r pool/users@today\fR
.fi
.in -2
.sp

.LP
\fBExample 17 \fRSetting \fBsharenfs\fR Property Options on a ZFS File System
.sp
.LP
The following commands show how to set \fBsharenfs\fR property options to
enable \fBrw\fR access for a set of \fBIP\fR addresses and to enable root
access for system \fBneo\fR on the \fBtank/home\fR file system.

.sp
.in +2
.nf
# \fBzfs set sharenfs='rw=@123.123.0.0/16,root=neo' tank/home\fR
.fi
.in -2
.sp

.sp
.LP
If you are using \fBDNS\fR for host name resolution, specify the fully
qualified hostname.

.LP
\fBExample 18 \fRDelegating ZFS Administration Permissions on a ZFS Dataset
.sp
.LP
The following example shows how to set permissions so that user \fBcindys\fR
can create, destroy, mount, and take snapshots on \fBtank/cindys\fR. The
permissions on \fBtank/cindys\fR are also displayed.
```



```
.sp
.in +2
.nf
# \fBzfs allow cindys create,destroy,mount,snapshot tank/cindys\fR
# \fBzfs allow tank/cindys\fR
-------------------------------------------------------------
Local+Descendent permissions on (tank/cindys)
          user cindys create,destroy,mount,snapshot
-------------------------------------------------------------
.fi
.in -2
.sp

.sp
.LP
Because the \fBtank/cindys\fR mount point permission is set to 755 by default,
user \fBcindys\fR will be unable to mount file systems under \fBtank/cindys\fR.
Set an \fBACL\fR similar to the following syntax to provide mount point access:
.sp
.in +2
.nf
# \fBchmod A+user:cindys:add_subdirectory:allow /tank/cindys\fR
.fi
.in -2
.sp

.LP
\fBExample 19 \fRDelegating Create Time Permissions on a ZFS Dataset
.sp
.LP
The following example shows how to grant anyone in the group \fBstaff\fR to
create file systems in \fBtank/users\fR. This syntax also allows staff members
to destroy their own file systems, but not destroy anyone else's file system.
The permissions on \fBtank/users\fR are also displayed.

.sp
.in +2
.nf
# \fBzfs allow staff create,mount tank/users\fR
# \fBzfs allow -c destroy tank/users\fR
# \fBzfs allow tank/users\fR
-------------------------------------------------------------
Create time permissions on (tank/users)
          create,destroy
Local+Descendent permissions on (tank/users)
          group staff create,mount
-------------------------------------------------------------
.fi
.in -2
.sp

.LP
\fBExample 20 \fRDefining and Granting a Permission Set on a ZFS Dataset
.sp
.LP
The following example shows how to define and grant a permission set on the
\fBtank/users\fR file system. The permissions on \fBtank/users\fR are also
displayed.

.sp
.in +2
.nf
# \fBzfs allow -s @pset create,destroy,snapshot,mount tank/users\fR
# \fBzfs allow staff @pset tank/users\fR
# \fBzfs allow tank/users\fR
-------------------------------------------------------------
```

```
3823 Permission sets on (tank/users)
3824         @pset create,destroy,mount,snapshot
3825 Create time permissions on (tank/users)
3826         create,destroy
3827 Local+Descendent permissions on (tank/users)
3828         group staff @pset,create,mount
3829 --------------------------------------------------------------
3830 .fi
3831 .in -2
3832 .sp

3834 .LP
3835 \fBExample 21 \fRDelegating Property Permissions on a ZFS Dataset
3836 .sp
3837 .LP
3838 The following example shows to grant the ability to set quotas and reservations
3839 on the \fBusers/home\fR file system. The permissions on \fBusers/home\fR are
3840 also displayed.

3842 .sp
3843 .in +2
3844 .nf
3845 # \fBzfs allow cindys quota,reservation users/home\fR
3846 # \fBzfs allow users/home\fR
3847 --------------------------------------------------------------
3848 Local+Descendent permissions on (users/home)
3849         user cindys quota,reservation
3850 --------------------------------------------------------------
3851 cindys% \fBzfs set quota=10G users/home/marks\fR
3852 cindys% \fBzfs get quota users/home/marks\fR
3853 NAME              PROPERTY  VALUE             SOURCE
3854 users/home/marks  quota     10G               local
3855 .fi
3856 .in -2
3857 .sp

3859 .LP
3860 \fBExample 22 \fRRemoving ZFS Delegated Permissions on a ZFS Dataset
3861 .sp
3862 .LP
3863 The following example shows how to remove the snapshot permission from the
3864 \fBstaff\fR group on the \fBtank/users\fR file system. The permissions on
3865 \fBtank/users\fR are also displayed.

3867 .sp
3868 .in +2
3869 .nf
3870 # \fBzfs unallow staff snapshot tank/users\fR
3871 # \fBzfs allow tank/users\fR
3872 --------------------------------------------------------------
3873 Permission sets on (tank/users)
3874         @pset create,destroy,mount,snapshot
3875 Create time permissions on (tank/users)
3876         create,destroy
3877 Local+Descendent permissions on (tank/users)
3878         group staff @pset,create,mount
3879 --------------------------------------------------------------
3880 .fi
3881 .in -2
3882 .sp

3884 .LP
3885 \fBExample 23\fR Showing the differences between a snapshot and a ZFS Dataset
3886 .sp
3887 .LP
3888 The following example shows how to see what has changed between a prior
```



```
3889 snapshot of a ZFS Dataset and its current state.  The \fB-F\fR option is used
3890 to indicate type information for the files affected.

3892 .sp
3893 .in +2
3894 .nf
3895 # zfs diff -F tank/test@before tank/test
3896 M       /       /tank/test/
3897 M       F       /tank/test/linked      (+1)
3898 R       F       /tank/test/oldname -> /tank/test/newname
3899 -       F       /tank/test/deleted
3900 +       F       /tank/test/created
3901 M       F       /tank/test/modified
3902 .fi
3903 .in -2
3904 .sp

3906 .SH EXIT STATUS
3907 .sp
3908 .LP
3909 The following exit values are returned:
3910 .sp
3911 .ne 2
3912 .na
3913 \fB\fB0\fR\fR
3914 .ad
3915 .sp .6
3916 .RS 4n
3917 Successful completion.
3918 .RE

3920 .sp
3921 .ne 2
3922 .na
3923 \fB\fB1\fR\fR
3924 .ad
3925 .sp .6
3926 .RS 4n
3927 An error occurred.
3928 .RE

3930 .sp
3931 .ne 2
3932 .na
3933 \fB\fB2\fR\fR
3934 .ad
3935 .sp .6
3936 .RS 4n
3937 Invalid command line options were specified.
3938 .RE

3940 .SH ATTRIBUTES
3941 .sp
3942 .LP
3943 See \fBattributes\fR(5) for descriptions of the following attributes:
3944 .sp

3946 .sp
3947 .TS
3948 box;
3949 c | c
3950 l | l .
3951 ATTRIBUTE TYPE  ATTRIBUTE VALUE
3952 _
3953 Interface Stability     Committed
3954 .TE
```

```
3956 .SH SEE ALSO
3957 .sp
3958 .LP
3959 \fBssh\fR(1), \fBiscsitadm\fR(1M), \fBmount\fR(1M), \fBshare\fR(1M),
3960 \fBsharemgr\fR(1M), \fBunshare\fR(1M), \fBzonecfg\fR(1M), \fBzpool\fR(1M),
3961 \fBchmod\fR(2), \fBstat\fR(2), \fBwrite\fR(2), \fBfsync\fR(3C),
3962 \fBdfstab\fR(4), \fBacl\fR(5), \fBattributes\fR(5)
3963 .sp
3964 .LP
3965 See the \fBgzip\fR(1) man page, which is not part of the SunOS man page
3966 collection.
3967 .sp
3968 .LP
3969 For information about using the \fBZFS\fR web-based management tool and other
3970 \fBZFS\fR features, see the \fISolaris ZFS Administration Guide\fR.
```

```
**********************************************************
    8109 Tue Jan 15 15:38:23 2013
new/usr/src/man/man5/zpool-features.5
%B
%B
**********************************************************
   1 '\" te
   2 .\" Copyright (c) 2012 by Delphix. All rights reserved.
   3 .\" Copyright (c) 2013 by Saso Kiselkov. All rights reserved.
   4 .\" The contents of this file are subject to the terms of the Common Development
   5 .\" and Distribution License (the "License").  You may not use this file except
   6 .\" in compliance with the License. You can obtain a copy of the license at
   7 .\" usr/src/OPENSOLARIS.LICENSE or http://www.opensolaris.org/os/licensing.
   8 .\"
   9 .\" See the License for the specific language governing permissions and
  10 .\" limitations under the License. When distributing Covered Code, include this
  11 .\" CDDL HEADER in each file and include the License file at
  12 .\" usr/src/OPENSOLARIS.LICENSE.  If applicable, add the following below this
  13 .\" CDDL HEADER, with the fields enclosed by brackets "[]" replaced with your
  14 .\" own identifying information:
  15 .\" Portions Copyright [yyyy] [name of copyright owner]
  16 .TH ZPOOL-FEATURES 5 "Mar 16, 2012"
  17 .SH NAME
  18 zpool\-features \- ZFS pool feature descriptions
  19 .SH DESCRIPTION
  20 .sp
  21 .LP
  22 ZFS pool on\-disk format versions are specified via "features" which replace
  23 the old on\-disk format numbers (the last supported on\-disk format number is
  24 28). To enable a feature on a pool use the \fBupgrade\fR subcommand of the
  25 \fBzpool\fR(1M) command, or set the \fBfeature@\fR\fIfeature_name\fR property
  26 to \fBenabled\fR.
  27 .sp
  28 .LP
  29 The pool format does not affect file system version compatibility or the ability
  30 to send file systems between pools.
  31 .sp
  32 .LP
  33 Since most features can be enabled independently of each other the on\-disk
  34 format of the pool is specified by the set of all features marked as
  35 \fBactive\fR on the pool. If the pool was created by another software version
  36 this set may include unsupported features.
  37 .SS "Identifying features"
  38 .sp
  39 .LP
  40 Every feature has a guid of the form \fIcom.example:feature_name\fR. The reverse
  41 DNS name ensures that the feature's guid is unique across all ZFS
  42 implementations. When unsupported features are encountered on a pool they will
  43 be identified by their guids. Refer to the documentation for the ZFS
  44 implementation that created the pool for information about those features.
  45 .sp
  46 .LP
  47 Each supported feature also has a short name. By convention a feature's short
  48 name is the portion of its guid which follows the ':' (e.g.
  49 \fIcom.example:feature_name\fR would have the short name \fIfeature_name\fR),
  50 however a feature's short name may differ across ZFS implementations if
  51 following the convention would result in name conflicts.
  52 .SS "Feature states"
  53 .sp
  54 .LP
  55 Features can be in one of three states:
  56 .sp
  57 .ne 2
  58 .na
  59 \fB\fBactive\fR\fR
  60 .ad
```



```
  61 .RS 12n
  62 This feature's on\-disk format changes are in effect on the pool. Support for
  63 this feature is required to import the pool in read\-write mode. If this
  64 feature is not read-only compatible, support is also required to import the pool
  65 in read\-only mode (see "Read\-only compatibility").
  66 .RE
  67
  68 .sp
  69 .ne 2
  70 .na
  71 \fB\fBenabled\fR\fR
  72 .ad
  73 .RS 12n
  74 An administrator has marked this feature as enabled on the pool, but the
  75 feature's on\-disk format changes have not been made yet. The pool can still be
  76 imported by software that does not support this feature, but changes may be made
  77 to the on\-disk format at any time which will move the feature to the
  78 \fBactive\fR state. Some features may support returning to the \fBenabled\fR
  79 state after becoming \fBactive\fR. See feature\-specific documentation for
  80 details.
  81 .RE
  82
  83 .sp
  84 .ne 2
  85 .na
  86 \fBdisabled\fR
  87 .ad
  88 .RS 12n
  89 This feature's on\-disk format changes have not been made and will not be made
  90 unless an administrator moves the feature to the \fBenabled\fR state. Features
  91 cannot be disabled once they have been enabled.
  92 .RE
  93
  94 .sp
  95 .LP
  96 The state of supported features is exposed through pool properties of the form
  97 \fIfeature@short_name\fR.
  98 .SS "Read\-only compatibility"
  99 .sp
 100 .LP
 101 Some features may make on\-disk format changes that do not interfere with other
 102 software's ability to read from the pool. These features are referred to as
 103 "read\-only compatible". If all unsupported features on a pool are read\-only
 104 compatible, the pool can be imported in read\-only mode by setting the
 105 \fBreadonly\fR property during import (see \fBzpool\fR(1M) for details on
 106 importing pools).
 107 .SS "Unsupported features"
 108 .sp
 109 .LP
 110 For each unsupported feature enabled on an imported pool a pool property
 111 named \fIunsupported@feature_guid\fR will indicate why the import was allowed
 112 despite the unsupported feature. Possible values for this property are:
 113
 114 .sp
 115 .ne 2
 116 .na
 117 \fB\fBinactive\fR\fR
 118 .ad
 119 .RS 12n
 120 The feature is in the \fBenabled\fR state and therefore the pool's on\-disk
 121 format is still compatible with software that does not support this feature.
 122 .RE
 123
 124 .sp
 125 .ne 2
 126 .na
```

```
\fB\fBreadonly\fR\fR
.ad
.RS 12n
The feature is read\-only compatible and the pool has been imported in
read\-only mode.
.RE

.SS "Feature dependencies"
.sp
.LP
Some features depend on other features being enabled in order to function
properly. Enabling a feature will automatically enable any features it
depends on.
.SH FEATURES
.sp
.LP
The following features are supported on this system:
.sp
.ne 2
.na
\fB\fBasync_destroy\fR\fR
.ad
.RS 4n
.TS
l l .
GUID    com.delphix:async_destroy
READ\-ONLY COMPATIBLE   yes
DEPENDENCIES    none
.TE

Destroying a file system requires traversing all of its data in order to
return its used space to the pool. Without \fBasync_destroy\fR the file system
is not fully removed until all space has been reclaimed. If the destroy
operation is interrupted by a reboot or power outage the next attempt to open
the pool will need to complete the destroy operation synchronously.

When \fBasync_destroy\fR is enabled the file system's data will be reclaimed
by a background process, allowing the destroy operation to complete without
traversing the entire file system. The background process is able to resume
interrupted destroys after the pool has been opened, eliminating the need
to finish interrupted destroys as part of the open operation. The amount
of space remaining to be reclaimed by the background process is available
through the \fBfreeing\fR property.

This feature is only \fBactive\fR while \fBfreeing\fR is non\-zero.
.RE

.sp
.ne 2
.na
\fB\fBempty_bpobj\fR\fR
.ad
.RS 4n
.TS
l l .
GUID    com.delphix:empty_bpobj
READ\-ONLY COMPATIBLE   yes
DEPENDENCIES    none
.TE

This feature increases the performance of creating and using a large
number of snapshots of a single filesystem or volume, and also reduces
the disk space required.

When there are many snapshots, each snapshot uses many Block Pointer
Objects (bpobj's) to track blocks associated with that snapshot.
```



```
However, in common use cases, most of these bpobj's are empty.  This
feature allows us to create each bpobj on-demand, thus eliminating the
empty bpobjs.

This feature is \fBactive\fR while there are any filesystems, volumes,
or snapshots which were created after enabling this feature.
.RE

.sp
.ne 2
.na
\fB\fBlz4_compress\fR\fR
.ad
.RS 4n
.TS
l l .
GUID    org.illumos:lz4_compress
READ\-ONLY COMPATIBLE   no
DEPENDENCIES    none
.TE

\fBlz4\fR is a high-performance real-time compression algorithm that
features significantly faster compression and decompression as well as a
higher compression ratio than the older \fBlzjb\fR compression.
Typically, \fBlz4\fR compression is approximately 50% faster on
compressible data and 200% faster on incompressible data than
\fBlzjb\fR. It is also approximately 80% faster on decompression, while
giving approximately 10% better compression ratio.

When the \fBlz4_compress\fR feature is set to \fBenabled\fR, the
administrator can turn on \fBlz4\fR compression on any dataset on the
pool using the \fBzfs\fR(1M) command. Please note that doing so will
immediately activate the \fBlz4_compress\fR feature on the underlying
pool (even before any data is written). Since this feature is not
read-only compatible, this operation will render the pool unimportable
on systems without support for the \fBlz4_compress\fR feature. At the
moment, this operation cannot be reversed. Booting off of
\fBlz4\fR-compressed root pools is supported.

.RE

.SH "SEE ALSO"
\fBzpool\fR(1M)
```

```
**********************************************************
   13405 Tue Jan 15 15:38:23 2013
new/usr/src/pkg/manifests/source-system-grub.mf
%B
**********************************************************
```

```
   1 #
   2 # CDDL HEADER START
   3 #
   4 # The contents of this file are subject to the terms of the
   5 # Common Development and Distribution License (the "License").
   6 # You may not use this file except in compliance with the License.
   7 #
   8 # You can obtain a copy of the license at usr/src/OPENSOLARIS.LICENSE
   9 # or http://www.opensolaris.org/os/licensing.
  10 # See the License for the specific language governing permissions
  11 # and limitations under the License.
  12 #
  13 # When distributing Covered Code, include this CDDL HEADER in each
  14 # file and include the License file at usr/src/OPENSOLARIS.LICENSE.
  15 # If applicable, add the following below this CDDL HEADER, with the
  16 # fields enclosed by brackets "[]" replaced with your own identifying
  17 # information: Portions Copyright [yyyy] [name of copyright owner]
  18 #
  19 # CDDL HEADER END
  20 #

  22 #
  23 # Copyright (c) 2010, Oracle and/or its affiliates. All rights reserved.
  24 #

  26 set name=pkg.fmri \
  27     value=pkg:/source/system/grub@0.97,$(PKGVERS_BUILTON)-$(PKGVERS_BRANCH)
  28 set name=pkg.description value="Source for GNU GRUB"
  29 set name=pkg.summary \
  30     value="Source for GNU GRUB - GNU GRand Unified Bootloader"
  31 set name=info.classification value=org.opensolaris.category.2008:System/Core
  32 set name=variant.arch value=i386
  33 dir path=usr group=sys
  34 dir path=usr/share
  35 dir path=usr/share/src group=sys
  36 dir path=usr/share/src/grub
  37 dir path=usr/share/src/grub/docs
  38 dir path=usr/share/src/grub/grub
  39 dir path=usr/share/src/grub/lib
  40 dir path=usr/share/src/grub/netboot
  41 dir path=usr/share/src/grub/stage1
  42 dir path=usr/share/src/grub/stage2
  43 dir path=usr/share/src/grub/stage2/zfs-include
  44 dir path=usr/share/src/grub/util
  45 file path=usr/share/src/grub/AUTHORS
  46 file path=usr/share/src/grub/BUGS
  47 file path=usr/share/src/grub/COPYING
  48 file path=usr/share/src/grub/ChangeLog
  49 file path=usr/share/src/grub/INSTALL
  50 file path=usr/share/src/grub/MAINTENANCE
  51 file path=usr/share/src/grub/Makefile.am
  52 file path=usr/share/src/grub/Makefile.in
  53 file path=usr/share/src/grub/NEWS
  54 file path=usr/share/src/grub/README
  55 file path=usr/share/src/grub/THANKS
  56 file path=usr/share/src/grub/TODO
  57 file path=usr/share/src/grub/acinclude.m4
  58 file path=usr/share/src/grub/aclocal.m4
  59 file path=usr/share/src/grub/compile mode=0755
  60 file path=usr/share/src/grub/config.guess mode=0755
  61 file path=usr/share/src/grub/config.h.in
```



```
  62 file path=usr/share/src/grub/config.sub mode=0755
  63 file path=usr/share/src/grub/configure mode=0755
  64 file path=usr/share/src/grub/configure.ac
  65 file path=usr/share/src/grub/depcomp mode=0755
  66 file path=usr/share/src/grub/docs/Makefile.am
  67 file path=usr/share/src/grub/docs/Makefile.in
  68 file path=usr/share/src/grub/docs/boot.S
  69 file path=usr/share/src/grub/docs/boot.S.texi
  70 file path=usr/share/src/grub/docs/grub-install.8
  71 file path=usr/share/src/grub/docs/grub-md5-crypt.8
  72 file path=usr/share/src/grub/docs/grub-terminfo.8
  73 file path=usr/share/src/grub/docs/grub.8
  74 file path=usr/share/src/grub/docs/grub.info
  75 file path=usr/share/src/grub/docs/grub.info-1
  76 file path=usr/share/src/grub/docs/grub.info-2
  77 file path=usr/share/src/grub/docs/grub.info-3
  78 file path=usr/share/src/grub/docs/grub.info-4
  79 file path=usr/share/src/grub/docs/grub.texi
  80 file path=usr/share/src/grub/docs/help2man
  81 file path=usr/share/src/grub/docs/internals.texi
  82 file path=usr/share/src/grub/docs/kernel.c
  83 file path=usr/share/src/grub/docs/kernel.c.texi
  84 file path=usr/share/src/grub/docs/mbchk.1
  85 file path=usr/share/src/grub/docs/mdate-sh
  86 file path=usr/share/src/grub/docs/menu.lst
  87 file path=usr/share/src/grub/docs/multiboot.h
  88 file path=usr/share/src/grub/docs/multiboot.h.texi
  89 file path=usr/share/src/grub/docs/multiboot.info
  90 file path=usr/share/src/grub/docs/multiboot.texi
  91 file path=usr/share/src/grub/docs/src2texi
  92 file path=usr/share/src/grub/docs/stamp-vti
  93 file path=usr/share/src/grub/docs/texinfo.tex
  94 file path=usr/share/src/grub/docs/version.texi
  95 file path=usr/share/src/grub/grub/Makefile.am
  96 file path=usr/share/src/grub/grub/Makefile.in
  97 file path=usr/share/src/grub/grub/asmstub.c
  98 file path=usr/share/src/grub/grub/main.c
  99 file path=usr/share/src/grub/install-sh mode=0755
 100 file path=usr/share/src/grub/lib/Makefile.am
 101 file path=usr/share/src/grub/lib/Makefile.in
 102 file path=usr/share/src/grub/lib/device.c
 103 file path=usr/share/src/grub/lib/device.h
 104 file path=usr/share/src/grub/lib/getopt.c
 105 file path=usr/share/src/grub/lib/getopt.h
 106 file path=usr/share/src/grub/lib/getopt1.c
 107 file path=usr/share/src/grub/missing mode=0755
 108 file path=usr/share/src/grub/mkinstalldirs mode=0755
 109 file path=usr/share/src/grub/netboot/$(ARCH)_byteswap.h
 110 file path=usr/share/src/grub/netboot/$(ARCH)_elf.h
 111 file path=usr/share/src/grub/netboot/$(ARCH)_endian.h
 112 file path=usr/share/src/grub/netboot/$(ARCH)_timer.c
 113 file path=usr/share/src/grub/netboot/3c595.c
 114 file path=usr/share/src/grub/netboot/3c595.h
 115 file path=usr/share/src/grub/netboot/3c90x.c
 116 file path=usr/share/src/grub/netboot/Makefile.am
 117 file path=usr/share/src/grub/netboot/Makefile.in
 118 file path=usr/share/src/grub/netboot/README.netboot
 119 file path=usr/share/src/grub/netboot/basemem.c
 120 file path=usr/share/src/grub/netboot/big_bswap.h
 121 file path=usr/share/src/grub/netboot/bootp.h
 122 file path=usr/share/src/grub/netboot/byteswap.h
 123 file path=usr/share/src/grub/netboot/config.c
 124 file path=usr/share/src/grub/netboot/cpu.h
 125 file path=usr/share/src/grub/netboot/davicom.c
 126 file path=usr/share/src/grub/netboot/dev.h
 127 file path=usr/share/src/grub/netboot/e1000.c
```

```
 128 file path=usr/share/src/grub/netboot/e1000_hw.h
 129 file path=usr/share/src/grub/netboot/eepro100.c
 130 file path=usr/share/src/grub/netboot/elf.h
 131 file path=usr/share/src/grub/netboot/endian.h
 132 file path=usr/share/src/grub/netboot/epic100.c
 133 file path=usr/share/src/grub/netboot/epic100.h
 134 file path=usr/share/src/grub/netboot/etherboot.h
 135 file path=usr/share/src/grub/netboot/forcedeth.c
 136 file path=usr/share/src/grub/netboot/fsys_tftp.c
 137 file path=usr/share/src/grub/netboot/grub.h
 138 file path=usr/share/src/grub/netboot/if_arp.h
 139 file path=usr/share/src/grub/netboot/if_ether.h
 140 file path=usr/share/src/grub/netboot/igmp.h
 141 file path=usr/share/src/grub/netboot/in.h
 142 file path=usr/share/src/grub/netboot/io.h
 143 file path=usr/share/src/grub/netboot/ip.h
 144 file path=usr/share/src/grub/netboot/isa.h
 145 file path=usr/share/src/grub/netboot/latch.h
 146 file path=usr/share/src/grub/netboot/little_bswap.h
 147 file path=usr/share/src/grub/netboot/mii.h
 148 file path=usr/share/src/grub/netboot/misc.c
 149 file path=usr/share/src/grub/netboot/natsemi.c
 150 file path=usr/share/src/grub/netboot/nfs.h
 151 file path=usr/share/src/grub/netboot/nic.c
 152 file path=usr/share/src/grub/netboot/nic.h
 153 file path=usr/share/src/grub/netboot/ns83820.c
 154 file path=usr/share/src/grub/netboot/ns8390.c
 155 file path=usr/share/src/grub/netboot/ns8390.h
 156 file path=usr/share/src/grub/netboot/osdep.h
 157 file path=usr/share/src/grub/netboot/pci.c
 158 file path=usr/share/src/grub/netboot/pci.h
 159 file path=usr/share/src/grub/netboot/pci_ids.h
 160 file path=usr/share/src/grub/netboot/pci_io.c
 161 file path=usr/share/src/grub/netboot/pcnet32.c
 162 file path=usr/share/src/grub/netboot/pic8259.c
 163 file path=usr/share/src/grub/netboot/pic8259.h
 164 file path=usr/share/src/grub/netboot/pnic.c
 165 file path=usr/share/src/grub/netboot/pnic_api.h
 166 file path=usr/share/src/grub/netboot/pxe.h
 167 file path=usr/share/src/grub/netboot/r8169.c
 168 file path=usr/share/src/grub/netboot/rtl8139.c
 169 file path=usr/share/src/grub/netboot/segoff.h
 170 file path=usr/share/src/grub/netboot/sis900.c
 171 file path=usr/share/src/grub/netboot/sis900.h
 172 file path=usr/share/src/grub/netboot/stdint.h
 173 file path=usr/share/src/grub/netboot/sundance.c
 174 file path=usr/share/src/grub/netboot/tftp.h
 175 file path=usr/share/src/grub/netboot/tg3.c
 176 file path=usr/share/src/grub/netboot/tg3.h
 177 file path=usr/share/src/grub/netboot/timer.c
 178 file path=usr/share/src/grub/netboot/timer.h
 179 file path=usr/share/src/grub/netboot/tlan.c
 180 file path=usr/share/src/grub/netboot/tlan.h
 181 file path=usr/share/src/grub/netboot/tulip.c
 182 file path=usr/share/src/grub/netboot/types.h
 183 file path=usr/share/src/grub/netboot/udp.h
 184 file path=usr/share/src/grub/netboot/undi.c
 185 file path=usr/share/src/grub/netboot/undi.h
 186 file path=usr/share/src/grub/netboot/via-rhine.c
 187 file path=usr/share/src/grub/netboot/w89c840.c
 188 file path=usr/share/src/grub/stage1/Makefile.am
 189 file path=usr/share/src/grub/stage1/Makefile.in
 190 file path=usr/share/src/grub/stage1/stage1.S
 191 file path=usr/share/src/grub/stage1/stage1.h
 192 file path=usr/share/src/grub/stage2/$(ARCH)-elf.h
 193 file path=usr/share/src/grub/stage2/Makefile.am
```



```
 194 file path=usr/share/src/grub/stage2/Makefile.in
 195 file path=usr/share/src/grub/stage2/apic.h
 196 file path=usr/share/src/grub/stage2/apm.S
 197 file path=usr/share/src/grub/stage2/asm.S
 198 file path=usr/share/src/grub/stage2/bios.c
 199 file path=usr/share/src/grub/stage2/boot.c
 200 file path=usr/share/src/grub/stage2/builtins.c
 201 file path=usr/share/src/grub/stage2/char_io.c
 202 file path=usr/share/src/grub/stage2/cmdline.c
 203 file path=usr/share/src/grub/stage2/common.c
 204 file path=usr/share/src/grub/stage2/console.c
 205 file path=usr/share/src/grub/stage2/defs.h
 206 file path=usr/share/src/grub/stage2/dir.h
 207 file path=usr/share/src/grub/stage2/disk_inode.h
 208 file path=usr/share/src/grub/stage2/disk_inode_ffs.h
 209 file path=usr/share/src/grub/stage2/disk_io.c
 210 file path=usr/share/src/grub/stage2/fat.h
 211 file path=usr/share/src/grub/stage2/filesys.h
 212 file path=usr/share/src/grub/stage2/freebsd.h
 213 file path=usr/share/src/grub/stage2/fs.h
 214 file path=usr/share/src/grub/stage2/fsys_ext2fs.c
 215 file path=usr/share/src/grub/stage2/fsys_fat.c
 216 file path=usr/share/src/grub/stage2/fsys_ffs.c
 217 file path=usr/share/src/grub/stage2/fsys_iso9660.c
 218 file path=usr/share/src/grub/stage2/fsys_jfs.c
 219 file path=usr/share/src/grub/stage2/fsys_minix.c
 220 file path=usr/share/src/grub/stage2/fsys_reiserfs.c
 221 file path=usr/share/src/grub/stage2/fsys_ufs.c
 222 file path=usr/share/src/grub/stage2/fsys_ufs2.c
 223 file path=usr/share/src/grub/stage2/fsys_vstafs.c
 224 file path=usr/share/src/grub/stage2/fsys_xfs.c
 225 file path=usr/share/src/grub/stage2/fsys_zfs.c
 226 file path=usr/share/src/grub/stage2/fsys_zfs.h
 227 file path=usr/share/src/grub/stage2/graphics.c
 228 file path=usr/share/src/grub/stage2/graphics.h
 229 file path=usr/share/src/grub/stage2/gunzip.c
 230 file path=usr/share/src/grub/stage2/hercules.c
 231 file path=usr/share/src/grub/stage2/hercules.h
 232 file path=usr/share/src/grub/stage2/imgact_aout.h
 233 file path=usr/share/src/grub/stage2/iso9660.h
 234 file path=usr/share/src/grub/stage2/jfs.h
 235 file path=usr/share/src/grub/stage2/mb_header.h
 236 file path=usr/share/src/grub/stage2/mb_info.h
 237 file path=usr/share/src/grub/stage2/md5.c
 238 file path=usr/share/src/grub/stage2/md5.h
 239 file path=usr/share/src/grub/stage2/nbi.h
 240 file path=usr/share/src/grub/stage2/nbloader.S
 241 file path=usr/share/src/grub/stage2/pc_slice.h
 242 file path=usr/share/src/grub/stage2/pxeloader.S
 243 file path=usr/share/src/grub/stage2/serial.c
 244 file path=usr/share/src/grub/stage2/serial.h
 245 file path=usr/share/src/grub/stage2/setjmp.S
 246 file path=usr/share/src/grub/stage2/shared.h
 247 file path=usr/share/src/grub/stage2/size_test
 248 file path=usr/share/src/grub/stage2/smp-imps.c
 249 file path=usr/share/src/grub/stage2/smp-imps.h
 250 file path=usr/share/src/grub/stage2/stage1_5.c
 251 file path=usr/share/src/grub/stage2/stage2.c
 252 file path=usr/share/src/grub/stage2/start.S
 253 file path=usr/share/src/grub/stage2/start_eltorito.S
 254 file path=usr/share/src/grub/stage2/term.h
 255 file path=usr/share/src/grub/stage2/terminfo.c
 256 file path=usr/share/src/grub/stage2/terminfo.h
 257 file path=usr/share/src/grub/stage2/tparm.c
 258 file path=usr/share/src/grub/stage2/tparm.h
 259 file path=usr/share/src/grub/stage2/ufs.h
```

```
260 file path=usr/share/src/grub/stage2/ufs2.h
261 file path=usr/share/src/grub/stage2/vstafs.h
262 file path=usr/share/src/grub/stage2/xfs.h
263 file path=usr/share/src/grub/stage2/zfs-include/dmu.h
264 file path=usr/share/src/grub/stage2/zfs-include/dmu_objset.h
265 file path=usr/share/src/grub/stage2/zfs-include/dnode.h
266 file path=usr/share/src/grub/stage2/zfs-include/dsl_dataset.h
267 file path=usr/share/src/grub/stage2/zfs-include/dsl_dir.h
268 file path=usr/share/src/grub/stage2/zfs-include/spa.h
269 file path=usr/share/src/grub/stage2/zfs-include/uberblock_impl.h
270 file path=usr/share/src/grub/stage2/zfs-include/vdev_impl.h
271 file path=usr/share/src/grub/stage2/zfs-include/zap_impl.h
272 file path=usr/share/src/grub/stage2/zfs-include/zap_leaf.h
273 file path=usr/share/src/grub/stage2/zfs-include/zfs.h
274 file path=usr/share/src/grub/stage2/zfs-include/zfs_acl.h
275 file path=usr/share/src/grub/stage2/zfs-include/zfs_znode.h
276 file path=usr/share/src/grub/stage2/zfs-include/zil.h
277 file path=usr/share/src/grub/stage2/zfs-include/zio.h
278 file path=usr/share/src/grub/stage2/zfs-include/zio_checksum.h
279 file path=usr/share/src/grub/stage2/zfs_fletcher.c
280 file path=usr/share/src/grub/stage2/zfs_lzjb.c
281 file path=usr/share/src/grub/stage2/zfs_lz4.c
282 file path=usr/share/src/grub/stage2/zfs_sha256.c
283 file path=usr/share/src/grub/stamp-h.in
284 file path=usr/share/src/grub/util/Makefile.am
285 file path=usr/share/src/grub/util/Makefile.in
286 file path=usr/share/src/grub/util/grub-image.in
287 file path=usr/share/src/grub/util/grub-install.in
288 file path=usr/share/src/grub/util/grub-md5-crypt.in
289 file path=usr/share/src/grub/util/grub-set-default.in
290 file path=usr/share/src/grub/util/grub-terminfo.in
291 file path=usr/share/src/grub/util/mbchk.c
292 file path=usr/share/src/grub/util/mkbimage
293 legacy pkg=SUNWgrubS desc="Source for GNU GRUB" \
294     name="Source for GNU GRUB - GNU GRand Unified Bootloader"
295 license cr_Sun license=cr_Sun
296 license usr/src/grub/grub-0.97/AUTHORS license=usr/src/grub/grub-0.97/AUTHORS
297 license usr/src/grub/grub-0.97/COPYING license=usr/src/grub/grub-0.97/COPYING
```

```
**********************************************************
    8285 Tue Jan 15 15:38:23 2013
new/usr/src/pkg/manifests/system-file-system-zfs.mf
%B
**********************************************************
```

```
   1 #
   2 # CDDL HEADER START
   3 #
   4 # The contents of this file are subject to the terms of the
   5 # Common Development and Distribution License (the "License").
   6 # You may not use this file except in compliance with the License.
   7 #
   8 # You can obtain a copy of the license at usr/src/OPENSOLARIS.LICENSE
   9 # or http://www.opensolaris.org/os/licensing.
  10 # See the License for the specific language governing permissions
  11 # and limitations under the License.
  12 #
  13 # When distributing Covered Code, include this CDDL HEADER in each
  14 # file and include the License file at usr/src/OPENSOLARIS.LICENSE.
  15 # If applicable, add the following below this CDDL HEADER, with the
  16 # fields enclosed by brackets "[]" replaced with your own identifying
  17 # information: Portions Copyright [yyyy] [name of copyright owner]
  18 #
  19 # CDDL HEADER END
  20 #

  22 #
  23 # Copyright (c) 2010, Oracle and/or its affiliates. All rights reserved.
  24 # Copyright (c) 2012 by Delphix. All rights reserved.
  25 # Copyright (c) 2013 by Saso Kiselkov. All rights reserved.
  26 #

  28 set name=pkg.fmri value=pkg:/system/file-system/zfs@$(PKGVERS)
  29 set name=pkg.description value="ZFS libraries and commands"
  30 set name=pkg.summary value=ZFS
  31 set name=info.classification \
  32     value="org.opensolaris.category.2008:System/File System"

  34 #
  35 # Skip manifest generation until build 134 can be required on the
  36 # build systems, due to a Python flavor identification bug in pkgdepend.
  37 #
  38 set name=org.opensolaris.nodepend value=true
  39 set name=variant.arch value=$(ARCH)
  40 dir path=etc group=sys
  41 dir path=etc/fs group=sys
  42 dir path=etc/fs/zfs group=sys
  43 dir path=etc/sysevent group=sys
  44 dir path=etc/sysevent/config group=sys
  45 dir path=etc/zfs group=sys
  46 dir path=kernel group=sys
  47 dir path=kernel/drv group=sys
  48 dir path=kernel/drv/$(ARCH64) group=sys
  49 dir path=kernel/fs group=sys
  50 dir path=kernel/fs/$(ARCH64) group=sys
  51 dir path=kernel/kmdb group=sys
  52 dir path=kernel/kmdb/$(ARCH64) group=sys
  53 dir path=lib
  54 dir path=lib/$(ARCH64)
  55 dir path=sbin group=sys
  56 dir path=usr group=sys
  57 dir path=usr/lib
  58 dir path=usr/lib/$(ARCH64)
  59 dir path=usr/lib/devfsadm group=sys
  60 dir path=usr/lib/devfsadm/linkmod group=sys
  61 dir path=usr/lib/fs group=sys
```



```
  62 dir path=usr/lib/fs/zfs group=sys
  63 dir path=usr/lib/mdb group=sys
  64 dir path=usr/lib/mdb/kvm group=sys
  65 dir path=usr/lib/mdb/kvm/$(ARCH64) group=sys
  66 dir path=usr/lib/mdb/proc group=sys
  67 $(sparc_ONLY)dir path=usr/lib/mdb/proc/$(ARCH64) group=sys
  68 $(i386_ONLY)dir path=usr/lib/mdb/proc/$(ARCH64)
  69 dir path=usr/lib/python2.6
  70 dir path=usr/lib/python2.6/vendor-packages
  71 dir path=usr/lib/python2.6/vendor-packages/zfs
  72 dir path=usr/lib/sysevent
  73 dir path=usr/lib/sysevent/modules
  74 dir path=usr/lib/zfs
  75 dir path=usr/sbin
  76 $(i386_ONLY)dir path=usr/sbin/$(ARCH32)
  77 dir path=usr/sbin/$(ARCH64)
  78 dir path=usr/share/man/man1m
  79 driver name=zfs perms="* 0600 root sys" perms="zfs 0666 root sys"
  80 file \
  81     path=etc/sysevent/config/SUNW,EC_zfs,ESC_ZFS_bootfs_vdev_attach,sysevent.con
  82     group=sys
  83 file path=kernel/drv/$(ARCH64)/zfs group=sys
  84 $(i386_ONLY)file path=kernel/drv/zfs group=sys
  85 file path=kernel/drv/zfs.conf group=sys
  86 file path=kernel/kmdb/$(ARCH64)/zfs group=sys mode=0555
  87 $(i386_ONLY)file path=kernel/kmdb/zfs group=sys mode=0555
  88 file path=lib/$(ARCH64)/libzfs.so.1
  89 file path=lib/$(ARCH64)/libzfs_core.so.1
  90 file path=lib/$(ARCH64)/llib-lzfs.ln
  91 file path=lib/$(ARCH64)/llib-lzfs_core.ln
  92 file path=lib/libzfs.so.1
  93 file path=lib/libzfs_core.so.1
  94 file path=lib/llib-lzfs
  95 file path=lib/llib-lzfs.ln
  96 file path=lib/llib-lzfs_core
  97 file path=lib/llib-lzfs_core.ln
  98 file path=sbin/zfs mode=0555
  99 file path=sbin/zpool mode=0555
 100 file path=usr/lib/$(ARCH64)/libzfs_jni.so.1
 101 file path=usr/lib/$(ARCH64)/libzpool.so.1
 102 file path=usr/lib/devfsadm/linkmod/SUNW_zfs_link.so group=sys
 103 file path=usr/lib/fs/zfs/bootinstall mode=0555
 104 file path=usr/lib/fs/zfs/fstyp.so.1 mode=0555
 105 file path=usr/lib/libzfs_jni.so.1
 106 $(i386_ONLY)file path=usr/lib/libzpool.so.1
 107 file path=usr/lib/mdb/kvm/$(ARCH64)/zfs.so group=sys mode=0555
 108 $(i386_ONLY)file path=usr/lib/mdb/kvm/zfs.so group=sys mode=0555
 109 file path=usr/lib/mdb/proc/$(ARCH64)/libzpool.so group=sys mode=0555
 110 file path=usr/lib/mdb/proc/libzpool.so group=sys mode=0555
 111 file path=usr/lib/python2.6/vendor-packages/zfs/__init__.py
 112 file path=usr/lib/python2.6/vendor-packages/zfs/__init__.pyc
 113 file path=usr/lib/python2.6/vendor-packages/zfs/allow.py
 114 file path=usr/lib/python2.6/vendor-packages/zfs/allow.pyc
 115 file path=usr/lib/python2.6/vendor-packages/zfs/dataset.py
 116 file path=usr/lib/python2.6/vendor-packages/zfs/dataset.pyc
 117 file path=usr/lib/python2.6/vendor-packages/zfs/groupspace.py
 118 file path=usr/lib/python2.6/vendor-packages/zfs/groupspace.pyc
 119 file path=usr/lib/python2.6/vendor-packages/zfs/holds.py
 120 file path=usr/lib/python2.6/vendor-packages/zfs/holds.pyc
 121 file path=usr/lib/python2.6/vendor-packages/zfs/ioctl.so
 122 file path=usr/lib/python2.6/vendor-packages/zfs/table.py
 123 file path=usr/lib/python2.6/vendor-packages/zfs/table.pyc
 124 file path=usr/lib/python2.6/vendor-packages/zfs/unallow.py
 125 file path=usr/lib/python2.6/vendor-packages/zfs/unallow.pyc
 126 file path=usr/lib/python2.6/vendor-packages/zfs/userspace.py
 127 file path=usr/lib/python2.6/vendor-packages/zfs/userspace.pyc
```

```
128 file path=usr/lib/python2.6/vendor-packages/zfs/util.py
129 file path=usr/lib/python2.6/vendor-packages/zfs/util.pyc
130 file path=usr/lib/sysevent/modules/zfs_mod.so group=sys
131 file path=usr/lib/zfs/availdevs mode=0555
132 file path=usr/lib/zfs/pyzfs.py mode=0555
133 file path=usr/lib/zfs/pyzfs.pyc mode=0555
134 $(i386_ONLY)file path=usr/sbin/$(ARCH32)/zdb mode=0555
135 file path=usr/sbin/$(ARCH64)/zdb mode=0555
136 file path=usr/sbin/zstreamdump mode=0555
137 file path=usr/share/man/man1m/zdb.1m
138 file path=usr/share/man/man1m/zfs.1m
139 file path=usr/share/man/man1m/zpool.1m
140 file path=usr/share/man/man1m/zstreamdump.1m
141 file path=usr/share/man/man5/zpool-features.5
142 hardlink path=kernel/fs/$(ARCH64)/zfs target=../../../kernel/drv/$(ARCH64)/zfs
143 $(i386_ONLY)hardlink path=kernel/fs/zfs target=../../kernel/drv/zfs
144 hardlink path=usr/lib/fs/zfs/fstyp target=../../../sbin/fstyp
145 hardlink path=usr/sbin/zdb target=../../usr/lib/isaexec
146 legacy pkg=SUNWzfskr desc="ZFS kernel root components" \
147     name="ZFS Kernel (Root)"
148 legacy pkg=SUNWzfsr desc="ZFS root components" name="ZFS (Root)"
149 legacy pkg=SUNWzfsu desc="ZFS libraries and commands" name="ZFS (Usr)"
150 license cr_Sun license=cr_Sun
151 license lic_CDDL license=lic_CDDL
152 license usr/src/uts/common/fs/zfs/THIRDPARTYLICENSE.lz4 \
153     license=usr/src/uts/common/fs/zfs/THIRDPARTYLICENSE.lz4
154 link path=etc/fs/zfs/mount target=../../../sbin/zfs
155 link path=etc/fs/zfs/umount target=../../../sbin/zfs
156 link path=lib/$(ARCH64)/libzfs.so target=libzfs.so.1
157 link path=lib/$(ARCH64)/libzfs_core.so target=libzfs_core.so.1
158 link path=lib/libzfs.so target=libzfs.so.1
159 link path=lib/libzfs_core.so target=libzfs_core.so.1
160 link path=usr/lib/$(ARCH64)/libzfs.so \
161     target=../../../lib/$(ARCH64)/libzfs.so.1
162 link path=usr/lib/$(ARCH64)/libzfs.so.1 \
163     target=../../../lib/$(ARCH64)/libzfs.so.1
164 link path=usr/lib/$(ARCH64)/libzfs_core.so \
165     target=../../../lib/$(ARCH64)/libzfs_core.so.1
166 link path=usr/lib/$(ARCH64)/libzfs_core.so.1 \
167     target=../../../lib/$(ARCH64)/libzfs_core.so.1
168 link path=usr/lib/$(ARCH64)/libzfs_jni.so target=libzfs_jni.so.1
169 link path=usr/lib/$(ARCH64)/libzpool.so target=libzpool.so.1
170 link path=usr/lib/$(ARCH64)/llib-lzfs.ln \
171     target=../../../lib/$(ARCH64)/llib-lzfs.ln
172 link path=usr/lib/$(ARCH64)/llib-lzfs_core.ln \
173     target=../../../lib/$(ARCH64)/llib-lzfs_core.ln
174 link path=usr/lib/fs/zfs/mount target=../../../../sbin/zfs
175 link path=usr/lib/fs/zfs/umount target=../../../../sbin/zfs
176 link path=usr/lib/libzfs.so target=../../lib/libzfs.so.1
177 link path=usr/lib/libzfs.so.1 target=../../lib/libzfs.so.1
178 link path=usr/lib/libzfs_core.so target=../../lib/libzfs_core.so.1
179 link path=usr/lib/libzfs_core.so.1 target=../../lib/libzfs_core.so.1
180 link path=usr/lib/libzfs_jni.so target=libzfs_jni.so.1
181 $(i386_ONLY)link path=usr/lib/libzpool.so target=libzpool.so.1
182 link path=usr/lib/llib-lzfs target=../../lib/llib-lzfs
183 link path=usr/lib/llib-lzfs.ln target=../../lib/llib-lzfs.ln
184 link path=usr/lib/llib-lzfs_core target=../../lib/llib-lzfs_core
185 link path=usr/lib/llib-lzfs_core.ln target=../../lib/llib-lzfs_core.ln
186 link path=usr/sbin/zfs target=../../sbin/zfs
187 link path=usr/sbin/zpool target=../../sbin/zpool
188 depend fmri=runtime/python-26 type=require
```

```
**********************************************************
   43097 Tue Jan 15 15:38:23 2013
new/usr/src/uts/common/Makefile.files
%B
**********************************************************
#
# CDDL HEADER START
#
# The contents of this file are subject to the terms of the
# Common Development and Distribution License (the "License").
# You may not use this file except in compliance with the License.
#
# You can obtain a copy of the license at usr/src/OPENSOLARIS.LICENSE
# or http://www.opensolaris.org/os/licensing.
# See the License for the specific language governing permissions
# and limitations under the License.
#
# When distributing Covered Code, include this CDDL HEADER in each
# file and include the License file at usr/src/OPENSOLARIS.LICENSE.
# If applicable, add the following below this CDDL HEADER, with the
# fields enclosed by brackets "[]" replaced with your own identifying
# information: Portions Copyright [yyyy] [name of copyright owner]
#
# CDDL HEADER END
#

#
# Copyright (c) 1991, 2010, Oracle and/or its affiliates. All rights reserved.
# Copyright (c) 2012 Nexenta Systems, Inc. All rights reserved.
# Copyright (c) 2012 by Delphix. All rights reserved.
# Copyright (c) 2013 by Saso Kiselkov. All rights reserved.
#

#
# This Makefile defines all file modules for the directory uts/common
# and its children. These are the source files which may be considered
# common to all SunOS systems.

i386_CORE_OBJS += \
                atomic.o        \
                avintr.o        \
                pic.o

sparc_CORE_OBJS +=

COMMON_CORE_OBJS +=             \
                beep.o          \
                bitset.o        \
                bp_map.o        \
                brand.o         \
                cpucaps.o       \
                cmt.o           \
                cmt_policy.o    \
                cpu.o           \
                cpu_event.o     \
                cpu_intr.o      \
                cpu_pm.o        \
                cpupart.o       \
                cap_util.o      \
                disp.o          \
                group.o         \
                kstat_fr.o      \
                iscsiboot_prop.o        \
                lgrp.o          \
                lgrp_topo.o     \
                mmapobj.o       \
```



```
                mutex.o         \
                page_lock.o     \
                page_retire.o   \
                panic.o         \
                param.o         \
                pg.o            \
                pghw.o          \
                putnext.o       \
                rctl_proc.o     \
                rwlock.o        \
                seg_kmem.o      \
                softint.o       \
                string.o        \
                strtol.o        \
                strtoul.o       \
                strtoll.o       \
                strtoull.o      \
                thread_intr.o   \
                vm_page.o       \
                vm_pagelist.o   \
                zlib_obj.o      \
                clock_tick.o

CORE_OBJS +=    $(COMMON_CORE_OBJS) $($(MACH)_CORE_OBJS)

ZLIB_OBJS =     zutil.o zmod.o zmod_subr.o \
                adler32.o crc32.o deflate.o inffast.o \
                inflate.o inftrees.o trees.o

GENUNIX_OBJS += \
                access.o        \
                acl.o           \
                acl_common.o    \
                adjtime.o       \
                alarm.o         \
                aio_subr.o      \
                auditsys.o      \
                audit_core.o    \
                audit_zone.o    \
                audit_memory.o  \
                autoconf.o      \
                avl.o           \
                bdev_dsort.o    \
                bio.o           \
                bitmap.o        \
                blabel.o        \
                brandsys.o      \
                bz2blocksort.o  \
                bz2compress.o   \
                bz2decompress.o \
                bz2randtable.o  \
                bz2bzlib.o      \
                bz2crctable.o   \
                bz2huffman.o    \
                callb.o         \
                callout.o       \
                chdir.o         \
                chmod.o         \
                chown.o         \
                cladm.o         \
                class.o         \
                clock.o         \
                clock_highres.o \
                clock_realtime.o\
                close.o         \
                compress.o      \
```

```
128                 condvar.o       \
129                 conf.o          \
130                 console.o       \
131                 contract.o      \
132                 copyops.o       \
133                 core.o          \
134                 corectl.o       \
135                 cred.o          \
136                 cs_stubs.o      \
137                 dacf.o          \
138                 dacf_clnt.o     \
139                 damap.o \
140                 cyclic.o        \
141                 ddi.o           \
142                 ddifm.o         \
143                 ddi_hp_impl.o   \
144                 ddi_hp_ndi.o    \
145                 ddi_intr.o      \
146                 ddi_intr_impl.o \
147                 ddi_intr_irm.o  \
148                 ddi_nodeid.o    \
149                 ddi_timer.o     \
150                 devcfg.o        \
151                 devcache.o      \
152                 device.o        \
153                 devid.o         \
154                 devid_cache.o   \
155                 devid_scsi.o    \
156                 devid_smp.o     \
157                 devpolicy.o     \
158                 disp_lock.o     \
159                 dnlc.o          \
160                 driver.o        \
161                 dumpsubr.o      \
162                 driver_lyr.o    \
163                 dtrace_subr.o   \
164                 errorq.o        \
165                 etheraddr.o     \
166                 evchannels.o    \
167                 exacct.o        \
168                 exacct_core.o   \
169                 exec.o          \
170                 exit.o          \
171                 fbio.o          \
172                 fcntl.o         \
173                 fdbuffer.o      \
174                 fdsync.o        \
175                 fem.o           \
176                 ffs.o           \
177                 fio.o           \
178                 flock.o         \
179                 fm.o            \
180                 fork.o          \
181                 vpm.o           \
182                 fs_reparse.o    \
183                 fs_subr.o       \
184                 fsflush.o       \
185                 ftrace.o        \
186                 getcwd.o        \
187                 getdents.o      \
188                 getloadavg.o    \
189                 getpagesizes.o  \
190                 getpid.o        \
191                 gfs.o           \
192                 rusagesys.o     \
193                 gid.o           \
```



```
194                 groups.o        \
195                 grow.o          \
196                 hat_refmod.o    \
197                 id32.o          \
198                 id_space.o      \
199                 inet_ntop.o     \
200                 instance.o      \
201                 ioctl.o         \
202                 ip_cksum.o      \
203                 issetugid.o     \
204                 ippconf.o       \
205                 kcpc.o          \
206                 kdi.o           \
207                 kiconv.o        \
208                 klpd.o          \
209                 kmem.o          \
210                 ksyms_snapshot.o        \
211                 l_strplumb.o    \
212                 labelsys.o      \
213                 link.o          \
214                 list.o          \
215                 lockstat_subr.o \
216                 log_sysevent.o  \
217                 logsubr.o       \
218                 lookup.o        \
219                 lseek.o         \
220                 ltos.o          \
221                 lwp.o           \
222                 lwp_create.o    \
223                 lwp_info.o      \
224                 lwp_self.o      \
225                 lwp_sobj.o      \
226                 lwp_timer.o     \
227                 lwpsys.o        \
228                 main.o          \
229                 mmapobjsys.o    \
230                 memcntl.o       \
231                 memstr.o        \
232                 lgrpsys.o       \
233                 mkdir.o         \
234                 mknod.o         \
235                 mount.o         \
236                 move.o          \
237                 msacct.o        \
238                 multidata.o     \
239                 nbmlock.o       \
240                 ndifm.o         \
241                 nice.o          \
242                 netstack.o      \
243                 ntptime.o       \
244                 nvpair.o        \
245                 nvpair_alloc_system.o   \
246                 nvpair_alloc_fixed.o    \
247                 fnvpair.o       \
248                 octet.o         \
249                 open.o          \
250                 p_online.o      \
251                 pathconf.o      \
252                 pathname.o      \
253                 pause.o         \
254                 serializer.o    \
255                 pci_intr_lib.o  \
256                 pci_cap.o       \
257                 pcifm.o         \
258                 pgrp.o          \
259                 pgrpsys.o       \
```

```
260                 pid.o           \
261                 pkp_hash.o      \
262                 policy.o        \
263                 poll.o          \
264                 pool.o          \
265                 pool_pset.o     \
266                 port_subr.o     \
267                 ppriv.o         \
268                 printf.o        \
269                 priocntl.o      \
270                 priv.o          \
271                 priv_const.o    \
272                 proc.o          \
273                 procset.o       \
274                 processor_bind.o        \
275                 processor_info.o        \
276                 profil.o        \
277                 project.o       \
278                 qsort.o         \
279                 rctl.o          \
280                 rctlsys.o       \
281                 readlink.o      \
282                 refstr.o        \
283                 rename.o        \
284                 resolvepath.o   \
285                 retire_store.o  \
286                 process.o       \
287                 rlimit.o        \
288                 rmap.o          \
289                 rw.o            \
290                 rwstlock.o      \
291                 sad_conf.o      \
292                 sid.o           \
293                 sidsys.o        \
294                 sched.o         \
295                 schedctl.o      \
296                 sctp_crc32.o    \
297                 seg_dev.o       \
298                 seg_kp.o        \
299                 seg_kpm.o       \
300                 seg_map.o       \
301                 seg_vn.o        \
302                 seg_spt.o       \
303                 semaphore.o     \
304                 sendfile.o      \
305                 session.o       \
306                 share.o         \
307                 shuttle.o       \
308                 sig.o           \
309                 sigaction.o     \
310                 sigaltstack.o   \
311                 signotify.o     \
312                 sigpending.o    \
313                 sigprocmask.o   \
314                 sigqueue.o      \
315                 sigsendset.o    \
316                 sigsuspend.o    \
317                 sigtimedwait.o  \
318                 sleepq.o        \
319                 sock_conf.o     \
320                 space.o         \
321                 sscanf.o        \
322                 stat.o          \
323                 statfs.o        \
324                 statvfs.o       \
325                 stol.o          \
```



```
326                 str_conf.o      \
327                 strcalls.o      \
328                 stream.o        \
329                 streamio.o      \
330                 strext.o        \
331                 strsubr.o       \
332                 strsun.o        \
333                 subr.o          \
334                 sunddi.o        \
335                 sunmdi.o        \
336                 sunndi.o        \
337                 sunpci.o        \
338                 sunpm.o         \
339                 sundlpi.o       \
340                 suntpi.o        \
341                 swap_subr.o     \
342                 swap_vnops.o    \
343                 symlink.o       \
344                 sync.o          \
345                 sysclass.o      \
346                 sysconfig.o     \
347                 sysent.o        \
348                 sysfs.o         \
349                 systeminfo.o    \
350                 task.o          \
351                 taskq.o         \
352                 tasksys.o       \
353                 time.o          \
354                 timer.o         \
355                 times.o         \
356                 timers.o        \
357                 thread.o        \
358                 tlabel.o        \
359                 tnf_res.o       \
360                 turnstile.o     \
361                 tty_common.o    \
362                 u8_textprep.o   \
363                 uadmin.o        \
364                 uconv.o         \
365                 ucredsys.o      \
366                 uid.o           \
367                 umask.o         \
368                 umount.o        \
369                 uname.o         \
370                 unix_bb.o       \
371                 unlink.o        \
372                 urw.o           \
373                 utime.o         \
374                 utssys.o        \
375                 uucopy.o        \
376                 vfs.o           \
377                 vfs_conf.o      \
378                 vmem.o          \
379                 vm_anon.o       \
380                 vm_as.o         \
381                 vm_meter.o      \
382                 vm_pageout.o    \
383                 vm_pvn.o        \
384                 vm_rm.o         \
385                 vm_seg.o        \
386                 vm_subr.o       \
387                 vm_swap.o       \
388                 vm_usage.o      \
389                 vnode.o         \
390                 vuid_queue.o    \
391                 vuid_store.o    \
```

```
392                 waitq.o         \
393                 watchpoint.o    \
394                 yield.o         \
395                 scsi_confdata.o \
396                 xattr.o         \
397                 xattr_common.o  \
398                 xdr_mblk.o      \
399                 xdr_mem.o       \
400                 xdr.o           \
401                 xdr_array.o     \
402                 xdr_refer.o     \
403                 xhat.o          \
404                 zone.o

406 #
407 #       Stubs for the stand-alone linker/loader
408 #
409 sparc_GENSTUBS_OBJS =   \
410         kobj_stubs.o

412 i386_GENSTUBS_OBJS =

414 COMMON_GENSTUBS_OBJS =

416 GENSTUBS_OBJS += $(COMMON_GENSTUBS_OBJS) $($(MACH)_GENSTUBS_OBJS)

418 #
419 #       DTrace and DTrace Providers
420 #
421 DTRACE_OBJS += dtrace.o dtrace_isa.o dtrace_asm.o

423 SDT_OBJS += sdt_subr.o

425 PROFILE_OBJS += profile.o

427 SYSTRACE_OBJS += systrace.o

429 LOCKSTAT_OBJS += lockstat.o

431 FASTTRAP_OBJS += fasttrap.o fasttrap_isa.o

433 DCPC_OBJS += dcpc.o

435 #
436 #       Driver (pseudo-driver) Modules
437 #
438 IPP_OBJS +=     ippctl.o

440 AUDIO_OBJS += audio_client.o audio_ddi.o audio_engine.o \
441             audio_fltdata.o audio_format.o audio_ctrl.o \
442             audio_grc3.o audio_output.o audio_input.o \
443             audio_oss.o audio_sun.o

445 AUDIOEMU10K_OBJS += audioemu10k.o

447 AUDIOENS_OBJS += audioens.o

449 AUDIOVIA823X_OBJS += audiovia823x.o

451 AUDIOVIA97_OBJS += audiovia97.o

453 AUDIO1575_OBJS += audio1575.o

455 AUDIO810_OBJS += audio810.o

457 AUDIOCMI_OBJS += audiocmi.o
```



```
459 AUDIOCMIHD_OBJS += audiocmihd.o

461 AUDIOHD_OBJS += audiohd.o

463 AUDIOIXP_OBJS += audioixp.o

465 AUDIOLS_OBJS += audiols.o

467 AUDIOP16X_OBJS += audiop16x.o

469 AUDIOPCI_OBJS += audiopci.o

471 AUDIOSOLO_OBJS += audiosolo.o

473 AUDIOTS_OBJS += audiots.o

475 AC97_OBJS += ac97.o ac97_ad.o ac97_alc.o ac97_cmi.o

477 BLKDEV_OBJS += blkdev.o

479 CARDBUS_OBJS += cardbus.o cardbus_hp.o cardbus_cfg.o

481 CONSKBD_OBJS += conskbd.o

483 CONSMS_OBJS +=  consms.o

485 OLDPTY_OBJS +=  tty_ptyconf.o

487 PTC_OBJS +=     tty_pty.o

489 PTSL_OBJS +=    tty_pts.o

491 PTM_OBJS +=     ptm.o

493 MII_OBJS +=     mii.o mii_cicada.o mii_natsemi.o mii_intel.o mii_qualsemi.o \
494                 mii_marvell.o mii_realtek.o mii_other.o

496 PTS_OBJS +=     pts.o

498 PTY_OBJS +=     ptms_conf.o

500 SAD_OBJS +=     sad.o

502 MD4_OBJS +=     md4.o md4_mod.o

504 MD5_OBJS +=     md5.o md5_mod.o

506 SHA1_OBJS +=    sha1.o sha1_mod.o

508 SHA2_OBJS +=    sha2.o sha2_mod.o

510 IPGPC_OBJS +=   classifierddi.o classifier.o filters.o trie.o table.o \
511                 ba_table.o

513 DSCPMK_OBJS +=  dscpmk.o dscpmkddi.o

515 DLCOSMK_OBJS += dlcosmk.o dlcosmkddi.o

517 FLOWACCT_OBJS +=        flowacctddi.o flowacct.o

519 TOKENMT_OBJS += tokenmt.o tokenmtddi.o

521 TSWTCL_OBJS +=  tswtcl.o tswtclddi.o

523 ARP_OBJS +=     arpddi.o
```

```
525 ICMP_OBJS +=    icmpddi.o

527 ICMP6_OBJS +=   icmp6ddi.o

529 RTS_OBJS +=     rtsddi.o

531 IP_ICMP_OBJS =  icmp.o icmp_opt_data.o
532 IP_RTS_OBJS =   rts.o rts_opt_data.o
533 IP_TCP_OBJS =   tcp.o tcp_fusion.o tcp_opt_data.o tcp_sack.o tcp_stats.o \
534                 tcp_misc.o tcp_timers.o tcp_time_wait.o tcp_tpi.o tcp_output.o \
535                 tcp_input.o tcp_socket.o tcp_bind.o tcp_cluster.o tcp_tunables.o
536 IP_UDP_OBJS =   udp.o udp_opt_data.o udp_tunables.o udp_stats.o
537 IP_SCTP_OBJS =  sctp.o sctp_opt_data.o sctp_output.o \
538                 sctp_init.o sctp_input.o sctp_cookie.o \
539                 sctp_conn.o sctp_error.o sctp_snmp.o \
540                 sctp_tunables.o sctp_shutdown.o sctp_common.o \
541                 sctp_timer.o sctp_heartbeat.o sctp_hash.o \
542                 sctp_bind.o sctp_notify.o sctp_asconf.o \
543                 sctp_addr.o tn_ipopt.o tnet.o ip_netinfo.o \
544                 sctp_misc.o
545 IP_ILB_OBJS =   ilb.o ilb_nat.o ilb_conn.o ilb_alg_hash.o ilb_alg_rr.o

547 IP_OBJS +=      igmp.o ipmp.o ip.o ip6.o ip6_asp.o ip6_if.o ip6_ire.o \
548                 ip6_rts.o ip_if.o ip_ire.o ip_listutils.o ip_mroute.o \
549                 ip_multi.o ip2mac.o ip_ndp.o ip_rts.o ip_srcid.o \
550                 ipddi.o ipdrop.o mi.o nd.o tunables.o optcom.o snmpcom.o \
551                 ipsec_loader.o spd.o ipclassifier.o inet_common.o ip_squeue.o \
552                 squeue.o ip_sadb.o ip_ftable.o proto_set.o radix.o ip_dummy.o \
553                 ip_helper_stream.o ip_tunables.o \
554                 ip_output.o ip_input.o ip6_input.o ip6_output.o ip_arp.o \
555                 conn_opt.o ip_attr.o ip_dce.o \
556                 $(IP_ICMP_OBJS) \
557                 $(IP_RTS_OBJS) \
558                 $(IP_TCP_OBJS) \
559                 $(IP_UDP_OBJS) \
560                 $(IP_SCTP_OBJS) \
561                 $(IP_ILB_OBJS)

563 IP6_OBJS +=     ip6ddi.o

565 HOOK_OBJS +=    hook.o

567 NETI_OBJS +=    neti_impl.o neti_mod.o neti_stack.o

569 KEYSOCK_OBJS += keysockddi.o keysock.o keysock_opt_data.o

571 IPNET_OBJS +=   ipnet.o ipnet_bpf.o

573 SPDSOCK_OBJS += spdsockddi.o spdsock.o spdsock_opt_data.o

575 IPSECESP_OBJS += ipsecespddi.o ipsecesp.o

577 IPSECAH_OBJS += ipsecahddi.o ipsecah.o sadb.o

579 SPPP_OBJS +=    sppp.o sppp_dlpi.o sppp_mod.o s_common.o

581 SPPPTUN_OBJS += sppptun.o sppptun_mod.o

583 SPPPASYN_OBJS += spppasyn.o spppasyn_mod.o

585 SPPPCOMP_OBJS += spppcomp.o spppcomp_mod.o deflate.o bsd-comp.o vjcompress.o \
586                 zlib.o

588 TCP_OBJS +=     tcpddi.o
```



```
590 TCP6_OBJS +=    tcp6ddi.o

592 NCA_OBJS +=     ncaddi.o

594 SDP_SOCK_MOD_OBJS += sockmod_sdp.o socksdp.o socksdpsubr.o

596 SCTP_SOCK_MOD_OBJS += sockmod_sctp.o socksctp.o socksctpsubr.o

598 PFP_SOCK_MOD_OBJS += sockmod_pfp.o

600 RDS_SOCK_MOD_OBJS += sockmod_rds.o

602 RDS_OBJS +=     rdsddi.o rdssubr.o rds_opt.o rds_ioctl.o

604 RDSIB_OBJS +=   rdsib.o rdsib_ib.o rdsib_cm.o rdsib_ep.o rdsib_buf.o \
605                 rdsib_debug.o rdsib_sc.o

607 RDSV3_OBJS +=   af_rds.o rdsv3_ddi.o bind.o loop.o threads.o connection.o \
608                 transport.o cong.o sysctl.o message.o rds_recv.o send.o \
609                 stats.o info.o page.o rdma_transport.o ib_ring.o ib_rdma.o \
610                 ib_recv.o ib.o ib_send.o ib_sysctl.o ib_stats.o ib_cm.o \
611                 rdsv3_sc.o rdsv3_debug.o rdsv3_impl.o rdma.o rdsv3_af_thr.o

613 ISER_OBJS +=    iser.o iser_cm.o iser_cq.o iser_ib.o iser_idm.o \
614                 iser_resource.o iser_xfer.o

616 UDP_OBJS +=     udpddi.o

618 UDP6_OBJS +=    udp6ddi.o

620 SY_OBJS +=      gentty.o

622 TCO_OBJS +=     ticots.o

624 TCOO_OBJS +=    ticotsord.o

626 TCL_OBJS +=     ticlts.o

628 TL_OBJS +=      tl.o

630 DUMP_OBJS +=    dump.o

632 BPF_OBJS +=     bpf.o bpf_filter.o bpf_mod.o bpf_dlt.o bpf_mac.o

634 CLONE_OBJS +=   clone.o

636 CN_OBJS +=      cons.o

638 DLD_OBJS +=     dld_drv.o dld_proto.o dld_str.o dld_flow.o

640 DLS_OBJS +=     dls.o dls_link.o dls_mod.o dls_stat.o dls_mgmt.o

642 GLD_OBJS +=     gld.o gldutil.o

644 MAC_OBJS +=     mac.o  mac_bcast.o mac_client.o mac_datapath_setup.o mac_flow.o
645                 mac_hio.o mac_mod.o mac_ndd.o mac_provider.o mac_sched.o \
646                 mac_protect.o mac_soft_ring.o mac_stat.o mac_util.o

648 MAC_6TO4_OBJS +=        mac_6to4.o

650 MAC_ETHER_OBJS +=       mac_ether.o

652 MAC_IPV4_OBJS +=        mac_ipv4.o

654 MAC_IPV6_OBJS +=        mac_ipv6.o
```

```
656 MAC_WIFI_OBJS +=        mac_wifi.o

658 MAC_IB_OBJS +=          mac_ib.o

660 IPTUN_OBJS +=   iptun_dev.o iptun_ctl.o iptun.o

662 AGGR_OBJS +=    aggr_dev.o aggr_ctl.o aggr_grp.o aggr_port.o \
663                 aggr_send.o aggr_recv.o aggr_lacp.o

665 SOFTMAC_OBJS += softmac_main.o softmac_ctl.o softmac_capab.o \
666                 softmac_dev.o softmac_stat.o softmac_pkt.o softmac_fp.o

668 NET80211_OBJS += net80211.o net80211_proto.o net80211_input.o \
669                  net80211_output.o net80211_node.o net80211_crypto.o \
670                  net80211_crypto_none.o net80211_crypto_wep.o net80211_ioctl.o \
671                  net80211_crypto_tkip.o net80211_crypto_ccmp.o  \
672                  net80211_ht.o

674 VNIC_OBJS +=    vnic_ctl.o vnic_dev.o

676 SIMNET_OBJS +=  simnet.o

678 IB_OBJS +=      ibnex.o ibnex_ioctl.o ibnex_hca.o

680 IBCM_OBJS +=    ibcm_impl.o ibcm_sm.o ibcm_ti.o ibcm_utils.o ibcm_path.o \
681                 ibcm_arp.o ibcm_arp_link.o

683 IBDM_OBJS +=    ibdm.o

685 IBDMA_OBJS +=   ibdma.o

687 IBMF_OBJS +=    ibmf.o ibmf_impl.o ibmf_dr.o ibmf_wqe.o ibmf_ud_dest.o ibmf_mod.
688                 ibmf_send.o ibmf_recv.o ibmf_handlers.o ibmf_trans.o \
689                 ibmf_timers.o ibmf_msg.o ibmf_utils.o ibmf_rmpp.o \
690                 ibmf_saa.o ibmf_saa_impl.o ibmf_saa_utils.o ibmf_saa_events.o

692 IBTL_OBJS +=    ibtl_impl.o ibtl_util.o ibtl_mem.o ibtl_handlers.o ibtl_qp.o \
693                 ibtl_cq.o ibtl_wr.o ibtl_hca.o ibtl_chan.o ibtl_cm.o \
694                 ibtl_mcg.o ibtl_ibnex.o ibtl_srq.o ibtl_part.o

696 TAVOR_OBJS +=   tavor.o tavor_agents.o tavor_cfg.o tavor_ci.o tavor_cmd.o \
697                 tavor_cq.o tavor_event.o tavor_ioctl.o tavor_misc.o \
698                 tavor_mr.o tavor_qp.o tavor_qpmod.o tavor_rsrc.o \
699                 tavor_srq.o tavor_stats.o tavor_umap.o tavor_wr.o

701 HERMON_OBJS +=  hermon.o hermon_agents.o hermon_cfg.o hermon_ci.o hermon_cmd.o \
702                 hermon_cq.o hermon_event.o hermon_ioctl.o hermon_misc.o \
703                 hermon_mr.o hermon_qp.o hermon_qpmod.o hermon_rsrc.o \
704                 hermon_srq.o hermon_stats.o hermon_umap.o hermon_wr.o \
705                 hermon_fcoib.o hermon_fm.o

707 DAPLT_OBJS +=   daplt.o

709 SOL_OFS_OBJS += sol_cma.o sol_ib_cma.o sol_uobj.o \
710                 sol_ofs_debug_util.o sol_ofs_gen_util.o \
711                 sol_kverbs.o

713 SOL_UCMA_OBJS +=        sol_ucma.o

715 SOL_UVERBS_OBJS +=      sol_uverbs.o sol_uverbs_comp.o sol_uverbs_event.o \
716                         sol_uverbs_hca.o sol_uverbs_qp.o

718 SOL_UMAD_OBJS += sol_umad.o

720 KSTAT_OBJS +=   kstat.o
```



```
722 KSYMS_OBJS +=   ksyms.o

724 INSTANCE_OBJS += inst_sync.o

726 IWSCN_OBJS +=   iwscons.o

728 LOFI_OBJS +=    lofi.o LzmaDec.o

730 FSSNAP_OBJS +=  fssnap.o

732 FSSNAPIF_OBJS += fssnap_if.o

734 MM_OBJS +=      mem.o

736 PHYSMEM_OBJS += physmem.o

738 OPTIONS_OBJS += options.o

740 WINLOCK_OBJS += winlockio.o

742 PM_OBJS +=      pm.o
743 SRN_OBJS +=     srn.o

745 PSEUDO_OBJS +=  pseudonex.o

747 RAMDISK_OBJS += ramdisk.o

749 LLC1_OBJS += llc1.o

751 USBKBM_OBJS += usbkbm.o

753 USBWCM_OBJS += usbwcm.o

755 BOFI_OBJS += bofi.o

757 HID_OBJS += hid.o

759 HWA_RC_OBJS += hwarc.o

761 USBSKEL_OBJS += usbskel.o

763 USBVC_OBJS += usbvc.o usbvc_v4l2.o

765 HIDPARSER_OBJS += hidparser.o

767 USB_AC_OBJS += usb_ac.o

769 USB_AS_OBJS += usb_as.o

771 USB_AH_OBJS += usb_ah.o

773 USBMS_OBJS += usbms.o

775 USBPRN_OBJS += usbprn.o

777 UGEN_OBJS += ugen.o

779 USBSER_OBJS += usbser.o usbser_rseq.o

781 USBSACM_OBJS += usbsacm.o

783 USBSER_KEYSPAN_OBJS += usbser_keyspan.o keyspan_dsd.o keyspan_pipe.o

785 USBS49_FW_OBJS += keyspan_49fw.o

787 USBSPRL_OBJS += usbser_pl2303.o pl2303_dsd.o
```

```
789 WUSB_CA_OBJS += wusb_ca.o

791 USBFTDI_OBJS += usbser_uftdi.o uftdi_dsd.o

793 USBECM_OBJS += usbecm.o

795 WC_OBJS += wscons.o vcons.o

797 VCONS_CONF_OBJS += vcons_conf.o

799 SCSI_OBJS +=    scsi_capabilities.o scsi_confsubr.o scsi_control.o \
800                 scsi_data.o scsi_fm.o scsi_hba.o scsi_reset_notify.o \
801                 scsi_resource.o scsi_subr.o scsi_transport.o scsi_watch.o \
802                 smp_transport.o

804 SCSI_VHCI_OBJS +=               scsi_vhci.o mpapi_impl.o scsi_vhci_tpgs.o

806 SCSI_VHCI_F_SYM_OBJS +=         sym.o

808 SCSI_VHCI_F_TPGS_OBJS +=        tpgs.o

810 SCSI_VHCI_F_ASYM_SUN_OBJS +=    asym_sun.o

812 SCSI_VHCI_F_SYM_HDS_OBJS +=     sym_hds.o

814 SCSI_VHCI_F_TAPE_OBJS +=        tape.o

816 SCSI_VHCI_F_TPGS_TAPE_OBJS +=   tpgs_tape.o

818 SGEN_OBJS +=    sgen.o

820 SMP_OBJS +=     smp.o

822 SATA_OBJS +=    sata.o

824 USBA_OBJS +=    hcdi.o  usba.o  usbai.o hubdi.o parser.o genconsole.o \
825                 usbai_pipe_mgmt.o usbai_req.o usbai_util.o usbai_register.o \
826                 usba_devdb.o usba10_calls.o usba_ugen.o whcdi.o wa.o
827 USBA_WITHOUT_WUSB_OBJS +=       hcdi.o  usba.o  usbai.o hubdi.o parser.o gencons
828                 usbai_pipe_mgmt.o usbai_req.o usbai_util.o usbai_register.o \
829                 usba_devdb.o usba10_calls.o usba_ugen.o

831 USBA10_OBJS +=  usba10.o

833 RSM_OBJS +=     rsm.o   rsmka_pathmanager.o     rsmka_util.o

835 RSMOPS_OBJS +=  rsmops.o

837 S1394_OBJS +=   t1394.o t1394_errmsg.o s1394.o s1394_addr.o s1394_asynch.o \
838                 s1394_bus_reset.o s1394_cmp.o s1394_csr.o s1394_dev_disc.o \
839                 s1394_fa.o s1394_fcp.o \
840                 s1394_hotplug.o s1394_isoch.o s1394_misc.o h1394.o nx1394.o

842 HCI1394_OBJS += hci1394.o hci1394_async.o hci1394_attach.o hci1394_buf.o \
843                 hci1394_csr.o hci1394_detach.o hci1394_extern.o \
844                 hci1394_ioctl.o hci1394_isoch.o hci1394_isr.o \
845                 hci1394_ixl_comp.o hci1394_ixl_isr.o hci1394_ixl_misc.o \
846                 hci1394_ixl_update.o hci1394_misc.o hci1394_ohci.o \
847                 hci1394_q.o hci1394_s1394if.o hci1394_tlabel.o \
848                 hci1394_tlist.o hci1394_vendor.o

850 AV1394_OBJS +=  av1394.o av1394_as.o av1394_async.o av1394_cfgrom.o \
851                 av1394_cmp.o av1394_fcp.o av1394_isoch.o av1394_isoch_chan.o \
852                 av1394_isoch_recv.o av1394_isoch_xmit.o av1394_list.o \
853                 av1394_queue.o
```



```
855 DCAM1394_OBJS += dcam.o dcam_frame.o dcam_param.o dcam_reg.o \
856                 dcam_ring_buff.o

858 SCSA1394_OBJS += hba.o sbp2_driver.o sbp2_bus.o

860 SBP2_OBJS += cfgrom.o sbp2.o

862 PMODEM_OBJS += pmodem.o pmodem_cis.o cis.o cis_callout.o cis_handlers.o cis_para

864 DSW_OBJS +=     dsw.o dsw_dev.o ii_tree.o

866 NCALL_OBJS +=   ncall.o \
867                 ncall_stub.o

869 RDC_OBJS +=     rdc.o \
870                 rdc_dev.o \
871                 rdc_io.o \
872                 rdc_clnt.o \
873                 rdc_prot_xdr.o \
874                 rdc_svc.o \
875                 rdc_bitmap.o \
876                 rdc_health.o \
877                 rdc_subr.o \
878                 rdc_diskq.o

880 RDCSRV_OBJS +=  rdcsrv.o

882 RDCSTUB_OBJS += rdc_stub.o

884 SDBC_OBJS +=    sd_bcache.o \
885                 sd_bio.o \
886                 sd_conf.o \
887                 sd_ft.o \
888                 sd_hash.o \
889                 sd_io.o \
890                 sd_misc.o \
891                 sd_pcu.o \
892                 sd_tdaemon.o \
893                 sd_trace.o \
894                 sd_iob_impl0.o \
895                 sd_iob_impl1.o \
896                 sd_iob_impl2.o \
897                 sd_iob_impl3.o \
898                 sd_iob_impl4.o \
899                 sd_iob_impl5.o \
900                 sd_iob_impl6.o \
901                 sd_iob_impl7.o \
902                 safestore.o \
903                 safestore_ram.o

905 NSCTL_OBJS +=   nsctl.o \
906                 nsc_cache.o \
907                 nsc_disk.o \
908                 nsc_dev.o \
909                 nsc_freeze.o \
910                 nsc_gen.o \
911                 nsc_mem.o \
912                 nsc_ncallio.o \
913                 nsc_power.o \
914                 nsc_resv.o \
915                 nsc_rmspin.o \
916                 nsc_solaris.o \
917                 nsc_trap.o \
918                 nsc_list.o
919 UNISTAT_OBJS += spuni.o \
```

```
 920                 spcs_s_k.o

 922 NSKERN_OBJS +=  nsc_ddi.o \
 923                 nsc_proc.o \
 924                 nsc_raw.o \
 925                 nsc_thread.o \
 926                 nskernd.o

 928 SV_OBJS +=      sv.o

 930 PMCS_OBJS += pmcs_attach.o pmcs_ds.o pmcs_intr.o pmcs_nvram.o pmcs_sata.o \
 931                 pmcs_scsa.o pmcs_smhba.o pmcs_subr.o pmcs_fwlog.o

 933 PMCS8001FW_C_OBJS +=    pmcs_fw_hdr.o
 934 PMCS8001FW_OBJS +=              $(PMCS8001FW_C_OBJS) SPCBoot.o ila.o firmware.o

 936 #
 937 #       Build up defines and paths.

 939 ST_OBJS +=      st.o    st_conf.o

 941 EMLXS_OBJS +=   emlxs_clock.o emlxs_dfc.o emlxs_dhchap.o emlxs_diag.o \
 942                 emlxs_download.o emlxs_dump.o emlxs_els.o emlxs_event.o \
 943                 emlxs_fcf.o emlxs_fcp.o emlxs_fct.o emlxs_hba.o emlxs_ip.o \
 944                 emlxs_mbox.o emlxs_mem.o emlxs_msg.o emlxs_node.o \
 945                 emlxs_pkt.o emlxs_sli3.o emlxs_sli4.o emlxs_solaris.o \
 946                 emlxs_thread.o

 948 EMLXS_FW_OBJS +=        emlxs_fw.o

 950 OCE_OBJS +=     oce_buf.o oce_fm.o oce_gld.o oce_hw.o oce_intr.o oce_main.o \
 951                 oce_mbx.o oce_mq.o oce_queue.o oce_rx.o oce_stat.o oce_tx.o \
 952                 oce_utils.o

 954 FCT_OBJS += discovery.o fct.o

 956 QLT_OBJS += 2400.o 2500.o 8100.o qlt.o qlt_dma.o

 958 SRPT_OBJS += srpt_mod.o srpt_ch.o srpt_cm.o srpt_ioc.o srpt_stp.o

 960 FCOE_OBJS += fcoe.o fcoe_eth.o fcoe_fc.o

 962 FCOET_OBJS += fcoet.o fcoet_eth.o fcoet_fc.o

 964 FCOEI_OBJS += fcoei.o fcoei_eth.o fcoei_lv.o

 966 ISCSIT_SHARED_OBJS += \
 967                 iscsit_common.o

 969 ISCSIT_OBJS +=  $(ISCSIT_SHARED_OBJS) \
 970                 iscsit.o iscsit_tgt.o iscsit_sess.o iscsit_login.o \
 971                 iscsit_text.o iscsit_isns.o iscsit_radiusauth.o \
 972                 iscsit_radiuspacket.o iscsit_auth.o iscsit_authclient.o

 974 PPPT_OBJS +=    alua_ic_if.o pppt.o pppt_msg.o pppt_tgt.o

 976 STMF_OBJS += lun_map.o stmf.o

 978 STMF_SBD_OBJS += sbd.o sbd_scsi.o sbd_pgr.o sbd_zvol.o

 980 SYSMSG_OBJS +=  sysmsg.o

 982 SES_OBJS +=     ses.o ses_sen.o ses_safte.o ses_ses.o

 984 TNF_OBJS +=     tnf_buf.o       tnf_trace.o     tnf_writer.o    trace_init.o \
 985                 trace_funcs.o   tnf_probe.o     tnf.o
```



```
 987 LOGINDMUX_OBJS += logindmux.o

 989 DEVINFO_OBJS += devinfo.o

 991 DEVPOLL_OBJS += devpoll.o

 993 DEVPOOL_OBJS += devpool.o

 995 I8042_OBJS +=   i8042.o

 997 KB8042_OBJS +=  \
 998                 at_keyprocess.o \
 999                 kb8042.o        \
1000                 kb8042_keytables.o

1002 MOUSE8042_OBJS += mouse8042.o

1004 FDC_OBJS +=     fdc.o

1006 ASY_OBJS +=     asy.o

1008 ECPP_OBJS +=    ecpp.o

1010 VUIDM3P_OBJS += vuidmice.o vuidm3p.o

1012 VUIDM4P_OBJS += vuidmice.o vuidm4p.o

1014 VUIDM5P_OBJS += vuidmice.o vuidm5p.o

1016 VUIDPS2_OBJS += vuidmice.o vuidps2.o

1018 HPCSVC_OBJS += hpcsvc.o

1020 PCIE_MISC_OBJS += pcie.o pcie_fault.o pcie_hp.o pciehpc.o pcishpc.o pcie_pwr.o p

1022 PCIHPNEXUS_OBJS += pcihp.o

1024 OPENEEPR_OBJS += openprom.o

1026 RANDOM_OBJS += random.o

1028 PSHOT_OBJS += pshot.o

1030 GEN_DRV_OBJS += gen_drv.o

1032 TCLIENT_OBJS += tclient.o

1034 TPHCI_OBJS += tphci.o

1036 TVHCI_OBJS += tvhci.o

1038 EMUL64_OBJS += emul64.o emul64_bsd.o

1040 FCP_OBJS += fcp.o

1042 FCIP_OBJS += fcip.o

1044 FCSM_OBJS += fcsm.o

1046 FCTL_OBJS += fctl.o

1048 FP_OBJS += fp.o

1050 QLC_OBJS += ql_api.o ql_debug.o ql_hba_fru.o ql_init.o ql_iocb.o ql_ioctl.o \
1051         ql_isr.o ql_mbx.o ql_nx.o ql_xioctl.o ql_fw_table.o
```

```
QLC_FW_2200_OBJS += ql_fw_2200.o

QLC_FW_2300_OBJS += ql_fw_2300.o

QLC_FW_2400_OBJS += ql_fw_2400.o

QLC_FW_2500_OBJS += ql_fw_2500.o

QLC_FW_6322_OBJS += ql_fw_6322.o

QLC_FW_8100_OBJS += ql_fw_8100.o

QLGE_OBJS += qlge.o qlge_dbg.o qlge_flash.o qlge_fm.o qlge_gld.o qlge_mpi.o

ZCONS_OBJS += zcons.o

NV_SATA_OBJS += nv_sata.o

SI3124_OBJS += si3124.o

AHCI_OBJS += ahci.o

PCIIDE_OBJS += pci-ide.o

PCEPP_OBJS += pcepp.o

CPC_OBJS += cpc.o

CPUID_OBJS += cpuid_drv.o

SYSEVENT_OBJS += sysevent.o

BL_OBJS += bl.o

DRM_OBJS += drm_sunmod.o drm_kstat.o drm_agpsupport.o \
            drm_auth.o drm_bufs.o drm_context.o drm_dma.o \
            drm_drawable.o drm_drv.o drm_fops.o drm_ioctl.o drm_irq.o \
            drm_lock.o drm_memory.o drm_msg.o drm_pci.o drm_scatter.o \
            drm_cache.o drm_gem.o drm_mm.o ati_pcigart.o

FM_OBJS += devfm.o devfm_machdep.o

RTLS_OBJS +=    rtls.o

#
#                       exec modules
#
AOUTEXEC_OBJS +=aout.o

ELFEXEC_OBJS += elf.o elf_notes.o old_notes.o

INTPEXEC_OBJS +=intp.o

SHBINEXEC_OBJS +=shbin.o

JAVAEXEC_OBJS +=java.o

#
#                       file system modules
#
AUTOFS_OBJS +=  auto_vfsops.o auto_vnops.o auto_subr.o auto_xdr.o auto_sys.o

CACHEFS_OBJS += cachefs_cnode.o         cachefs_cod.o \
                cachefs_dir.o           cachefs_dlog.o  cachefs_filegrp.o \
                cachefs_fscache.o       cachefs_ioctl.o cachefs_log.o \
```



```
                cachefs_module.o \
                cachefs_noopc.o         cachefs_resource.o \
                cachefs_strict.o \
                cachefs_subr.o          cachefs_vfsops.o \
                cachefs_vnops.o

DCFS_OBJS +=    dc_vnops.o

DEVFS_OBJS +=   devfs_subr.o    devfs_vfsops.o  devfs_vnops.o

DEV_OBJS  +=    sdev_subr.o     sdev_vfsops.o   sdev_vnops.o    \
                sdev_ptsops.o   sdev_zvolops.o  sdev_comm.o     \
                sdev_profile.o  sdev_ncache.o   sdev_netops.o   \
                sdev_ipnetops.o \
                sdev_vtops.o

CTFS_OBJS +=    ctfs_all.o ctfs_cdir.o ctfs_ctl.o ctfs_event.o \
                ctfs_latest.o ctfs_root.o ctfs_sym.o ctfs_tdir.o ctfs_tmpl.o

OBJFS_OBJS +=   objfs_vfs.o     objfs_root.o    objfs_common.o \
                objfs_odir.o    objfs_data.o

FDFS_OBJS +=    fdops.o

FIFO_OBJS +=    fifosubr.o      fifovnops.o

PIPE_OBJS +=    pipe.o

HSFS_OBJS +=    hsfs_node.o     hsfs_subr.o     hsfs_vfsops.o   hsfs_vnops.o \
                hsfs_susp.o     hsfs_rrip.o     hsfs_susp_subr.o

LOFS_OBJS +=    lofs_subr.o     lofs_vfsops.o   lofs_vnops.o

NAMEFS_OBJS +=  namevfs.o       namevno.o

NFS_OBJS +=     nfs_client.o    nfs_common.o    nfs_dump.o \
                nfs_subr.o      nfs_vfsops.o    nfs_vnops.o \
                nfs_xdr.o       nfs_sys.o       nfs_strerror.o \
                nfs3_vfsops.o   nfs3_vnops.o    nfs3_xdr.o \
                nfs_acl_vnops.o nfs_acl_xdr.o   nfs4_vfsops.o \
                nfs4_vnops.o    nfs4_xdr.o      nfs4_idmap.o \
                nfs4_shadow.o   nfs4_subr.o \
                nfs4_attr.o     nfs4_rnode.o    nfs4_client.o \
                nfs4_acache.o   nfs4_common.o   nfs4_client_state.o \
                nfs4_callback.o nfs4_recovery.o nfs4_client_secinfo.o \
                nfs4_client_debug.o     nfs_stats.o \
                nfs4_acl.o      nfs4_stub_vnops.o       nfs_cmd.o

NFSSRV_OBJS +=  nfs_server.o    nfs_srv.o       nfs3_srv.o \
                nfs_acl_srv.o   nfs_auth.o      nfs_auth_xdr.o \
                nfs_export.o    nfs_log.o       nfs_log_xdr.o \
                nfs4_srv.o      nfs4_state.o    nfs4_srv_attr.o \
                nfs4_srv_ns.o   nfs4_db.o       nfs4_srv_deleg.o \
                nfs4_deleg_ops.o nfs4_srv_readdir.o nfs4_dispatch.o

SMBSRV_SHARED_OBJS += \
                smb_inet.o \
                smb_match.o \
                smb_msgbuf.o \
                smb_oem.o \
                smb_string.o \
                smb_utf8.o \
                smb_door_legacy.o \
                smb_xdr.o \
                smb_token.o \
                smb_token_xdr.o \
```

```
1184                 smb_sid.o \
1185                 smb_native.o \
1186                 smb_netbios_util.o

1188 SMBSRV_OBJS +=  $(SMBSRV_SHARED_OBJS)                  \
1189                 smb_acl.o                              \
1190                 smb_alloc.o                            \
1191                 smb_close.o                            \
1192                 smb_common_open.o                      \
1193                 smb_common_transact.o                  \
1194                 smb_create.o                           \
1195                 smb_delete.o                           \
1196                 smb_directory.o                        \
1197                 smb_dispatch.o                         \
1198                 smb_echo.o                             \
1199                 smb_fem.o                              \
1200                 smb_find.o                             \
1201                 smb_flush.o                            \
1202                 smb_fsinfo.o                           \
1203                 smb_fsops.o                            \
1204                 smb_init.o                             \
1205                 smb_kdoor.o                            \
1206                 smb_kshare.o                           \
1207                 smb_kutil.o                            \
1208                 smb_lock.o                             \
1209                 smb_lock_byte_range.o                  \
1210                 smb_locking_andx.o                     \
1211                 smb_logoff_andx.o                      \
1212                 smb_mangle_name.o                      \
1213                 smb_mbuf_marshaling.o                  \
1214                 smb_mbuf_util.o                        \
1215                 smb_negotiate.o                        \
1216                 smb_net.o                              \
1217                 smb_node.o                             \
1218                 smb_nt_cancel.o                        \
1219                 smb_nt_create_andx.o                   \
1220                 smb_nt_transact_create.o               \
1221                 smb_nt_transact_ioctl.o                \
1222                 smb_nt_transact_notify_change.o        \
1223                 smb_nt_transact_quota.o                \
1224                 smb_nt_transact_security.o             \
1225                 smb_odir.o                             \
1226                 smb_ofile.o                            \
1227                 smb_open_andx.o                        \
1228                 smb_opipe.o                            \
1229                 smb_oplock.o                           \
1230                 smb_pathname.o                         \
1231                 smb_print.o                            \
1232                 smb_process_exit.o                     \
1233                 smb_query_fileinfo.o                   \
1234                 smb_read.o                             \
1235                 smb_rename.o                           \
1236                 smb_sd.o                               \
1237                 smb_seek.o                             \
1238                 smb_server.o                           \
1239                 smb_session.o                          \
1240                 smb_session_setup_andx.o               \
1241                 smb_set_fileinfo.o                     \
1242                 smb_signing.o                          \
1243                 smb_tree.o                             \
1244                 smb_trans2_create_directory.o          \
1245                 smb_trans2_dfs.o                       \
1246                 smb_trans2_find.o                      \
1247                 smb_tree_connect.o                     \
1248                 smb_unlock_byte_range.o                \
1249                 smb_user.o                             \
```



```
1250                 smb_vfs.o                              \
1251                 smb_vops.o                             \
1252                 smb_vss.o                              \
1253                 smb_write.o                            \
1254                 smb_write_raw.o

1256 PCFS_OBJS +=    pc_alloc.o      pc_dir.o        pc_node.o       pc_subr.o \
1257                 pc_vfsops.o     pc_vnops.o

1259 PROC_OBJS +=    prcontrol.o     prioctl.o       prsubr.o        prusrio.o \
1260                 prvfsops.o      prvnops.o

1262 MNTFS_OBJS +=   mntvfsops.o     mntvnops.o

1264 SHAREFS_OBJS += sharetab.o      sharefs_vfsops.o        sharefs_vnops.o

1266 SPEC_OBJS +=    specsubr.o      specvfsops.o    specvnops.o

1268 SOCK_OBJS +=    socksubr.o      sockvfsops.o    sockparams.o    \
1269                 socksyscalls.o  socktpi.o       sockstr.o \
1270                 sockcommon_vnops.o      sockcommon_subr.o \
1271                 sockcommon_sops.o       sockcommon.o    \
1272                 sock_notsupp.o  socknotify.o \
1273                 nl7c.o          nl7curi.o       nl7chttp.o      nl7clogd.o \
1274                 nl7cnca.o       sodirect.o      sockfilter.o

1276 TMPFS_OBJS +=   tmp_dir.o       tmp_subr.o      tmp_tnode.o     tmp_vfsops.o \
1277                 tmp_vnops.o

1279 UDFS_OBJS +=    udf_alloc.o     udf_bmap.o      udf_dir.o       \
1280                 udf_inode.o     udf_subr.o      udf_vfsops.o    \
1281                 udf_vnops.o

1283 UFS_OBJS +=     ufs_alloc.o     ufs_bmap.o      ufs_dir.o       ufs_xattr.o \
1284                 ufs_inode.o     ufs_subr.o      ufs_tables.o    ufs_vfsops.o \
1285                 ufs_vnops.o     quota.o         quotacalls.o    quota_ufs.o \
1286                 ufs_filio.o     ufs_lockfs.o    ufs_thread.o    ufs_trans.o \
1287                 ufs_acl.o       ufs_panic.o     ufs_directio.o  ufs_log.o \
1288                 ufs_extvnops.o  ufs_snap.o      lufs.o          lufs_thread.o \
1289                 lufs_log.o      lufs_map.o      lufs_top.o      lufs_debug.o
1290 VSCAN_OBJS +=   vscan_drv.o     vscan_svc.o vscan_door.o

1292 NSMB_OBJS +=    smb_conn.o      smb_dev.o       smb_iod.o       smb_pass.o \
1293                 smb_rq.o        smb_sign.o      smb_smb.o       smb_subrs.o \
1294                 smb_time.o      smb_tran.o      smb_trantcp.o   smb_usr.o \
1295                 subr_mchain.o

1297 SMBFS_COMMON_OBJS += smbfs_ntacl.o
1298 SMBFS_OBJS +=    smbfs_vfsops.o  smbfs_vnops.o   smbfs_node.o    \
1299                 smbfs_acl.o     smbfs_client.o  smbfs_smb.o     \
1300                 smbfs_subr.o    smbfs_subr2.o   \
1301                 smbfs_rwlock.o  smbfs_xattr.o   \
1302                 $(SMBFS_COMMON_OBJS)


1305 #
1306 #                       LVM modules
1307 #
1308 MD_OBJS += md.o md_error.o md_ioctl.o md_mddb.o md_names.o \
1309         md_med.o md_rename.o md_subr.o

1311 MD_COMMON_OBJS = md_convert.o md_crc.o md_revchk.o

1313 MD_DERIVED_OBJS = metamed_xdr.o meta_basic_xdr.o

1315 SOFTPART_OBJS += sp.o sp_ioctl.o
```

```
1317 STRIPE_OBJS += stripe.o stripe_ioctl.o

1319 HOTSPARES_OBJS += hotspares.o

1321 RAID_OBJS += raid.o raid_ioctl.o raid_replay.o raid_resync.o raid_hotspare.o

1323 MIRROR_OBJS += mirror.o mirror_ioctl.o mirror_resync.o

1325 NOTIFY_OBJS += md_notify.o

1327 TRANS_OBJS += mdtrans.o trans_ioctl.o trans_log.o

1329 ZFS_COMMON_OBJS +=             \
1330         arc.o                  \
1331         bplist.o               \
1332         bpobj.o                \
1333         bptree.o               \
1334         dbuf.o                 \
1335         ddt.o                  \
1336         ddt_zap.o              \
1337         dmu.o                  \
1338         dmu_diff.o             \
1339         dmu_send.o             \
1340         dmu_object.o           \
1341         dmu_objset.o           \
1342         dmu_traverse.o         \
1343         dmu_tx.o               \
1344         dnode.o                \
1345         dnode_sync.o           \
1346         dsl_dir.o              \
1347         dsl_dataset.o          \
1348         dsl_deadlist.o         \
1349         dsl_pool.o             \
1350         dsl_synctask.o         \
1351         dmu_zfetch.o           \
1352         dsl_deleg.o            \
1353         dsl_prop.o             \
1354         dsl_scan.o             \
1355         zfeature.o             \
1356         gzip.o                 \
1357         lz4.o                  \
1358         lz4hc.o                \
1359         lzjb.o                 \
1360         metaslab.o             \
1361         refcount.o             \
1362         sa.o                   \
1363         sha256.o               \
1364         spa.o                  \
1365         spa_config.o           \
1366         spa_errlog.o           \
1367         spa_history.o          \
1368         spa_misc.o             \
1369         space_map.o            \
1370         txg.o                  \
1371         uberblock.o            \
1372         unique.o               \
1373         vdev.o                 \
1374         vdev_cache.o           \
1375         vdev_file.o            \
1376         vdev_label.o           \
1377         vdev_mirror.o          \
1378         vdev_missing.o         \
1379         vdev_queue.o           \
1380         vdev_raidz.o           \
1381         vdev_root.o            \
```



```
1382         zap.o                  \
1383         zap_leaf.o             \
1384         zap_micro.o            \
1385         zfs_byteswap.o         \
1386         zfs_debug.o            \
1387         zfs_fm.o               \
1388         zfs_fuid.o             \
1389         zfs_sa.o               \
1390         zfs_znode.o            \
1391         zil.o                  \
1392         zio.o                  \
1393         zio_checksum.o         \
1394         zio_compress.o         \
1395         zio_inject.o           \
1396         zle.o                  \
1397         zrlock.o

1399 ZFS_SHARED_OBJS +=             \
1400         zfeature_common.o      \
1401         zfs_comutil.o          \
1402         zfs_deleg.o            \
1403         zfs_fletcher.o         \
1404         zfs_namecheck.o        \
1405         zfs_prop.o             \
1406         zpool_prop.o           \
1407         zprop_common.o

1409 ZFS_OBJS +=                    \
1410         $(ZFS_COMMON_OBJS)     \
1411         $(ZFS_SHARED_OBJS)     \
1412         vdev_disk.o            \
1413         zfs_acl.o              \
1414         zfs_ctldir.o           \
1415         zfs_dir.o              \
1416         zfs_ioctl.o            \
1417         zfs_log.o              \
1418         zfs_onexit.o           \
1419         zfs_replay.o           \
1420         zfs_rlock.o            \
1421         rrwlock.o              \
1422         zfs_vfsops.o           \
1423         zfs_vnops.o            \
1424         zvol.o

1426 ZUT_OBJS +=                    \
1427         zut.o

1429 #
1430 #                       streams modules
1431 #
1432 BUFMOD_OBJS     +=      bufmod.o

1434 CONNLD_OBJS +=  connld.o

1436 DEDUMP_OBJS +=  dedump.o

1438 DRCOMPAT_OBJS +=        drcompat.o

1440 LDLINUX_OBJS += ldlinux.o

1442 LDTERM_OBJS +=  ldterm.o uwidth.o

1444 PCKT_OBJS +=    pckt.o

1446 PFMOD_OBJS +=   pfmod.o
```

```
1448 PTEM_OBJS +=    ptem.o

1450 REDIRMOD_OBJS += strredirm.o

1452 TIMOD_OBJS +=   timod.o

1454 TIRDWR_OBJS +=  tirdwr.o

1456 TTCOMPAT_OBJS +=ttcompat.o

1458 LOG_OBJS +=     log.o

1460 PIPEMOD_OBJS += pipemod.o

1462 RPCMOD_OBJS +=  rpcmod.o        clnt_cots.o     clnt_clts.o \
1463                 clnt_gen.o      clnt_perr.o     mt_rpcinit.o    rpc_calmsg.o \
1464                 rpc_prot.o      rpc_sztypes.o   rpc_subr.o      rpcb_prot.o \
1465                 svc.o           svc_clts.o      svc_gen.o       svc_cots.o \
1466                 rpcsys.o        xdr_sizeof.o    clnt_rdma.o     svc_rdma.o \
1467                 xdr_rdma.o      rdma_subr.o     xdrrdma_sizeof.o

1469 TLIMOD_OBJS +=  tlimod.o        t_kalloc.o      t_kbind.o       t_kclose.o \
1470                 t_kconnect.o    t_kfree.o       t_kgtstate.o    t_kopen.o \
1471                 t_krcvudat.o    t_ksndudat.o    t_kspoll.o      t_kunbind.o \
1472                 t_kutil.o

1474 RLMOD_OBJS += rlmod.o

1476 TELMOD_OBJS += telmod.o

1478 CRYPTMOD_OBJS += cryptmod.o

1480 KB_OBJS +=      kbd.o           keytables.o

1482 #
1483 #                       ID mapping module
1484 #
1485 IDMAP_OBJS +=   idmap_mod.o     idmap_kapi.o    idmap_xdr.o     idmap_cache.o

1487 #
1488 #                       scheduling class modules
1489 #
1490 SDC_OBJS +=             sysdc.o

1492 RT_OBJS +=              rt.o
1493 RT_DPTBL_OBJS +=        rt_dptbl.o

1495 TS_OBJS +=              ts.o
1496 TS_DPTBL_OBJS +=        ts_dptbl.o

1498 IA_OBJS +=              ia.o

1500 FSS_OBJS +=             fss.o

1502 FX_OBJS +=              fx.o
1503 FX_DPTBL_OBJS +=        fx_dptbl.o

1505 #
1506 #                       Inter-Process Communication (IPC) modules
1507 #
1508 IPC_OBJS +=     ipc.o

1510 IPCMSG_OBJS +=  msg.o

1512 IPCSEM_OBJS +=  sem.o
```



```
1514 IPCSHM_OBJS +=  shm.o

1516 #
1517 #                       bignum module
1518 #
1519 COMMON_BIGNUM_OBJS += bignum_mod.o bignumimpl.o

1521 BIGNUM_OBJS += $(COMMON_BIGNUM_OBJS) $(BIGNUM_PSR_OBJS)

1523 #
1524 #                       kernel cryptographic framework
1525 #
1526 KCF_OBJS +=     kcf.o kcf_callprov.o kcf_cbufcall.o kcf_cipher.o kcf_crypto.o \
1527                 kcf_cryptoadm.o kcf_ctxops.o kcf_digest.o kcf_dual.o \
1528                 kcf_keys.o kcf_mac.o kcf_mech_tabs.o kcf_miscapi.o \
1529                 kcf_object.o kcf_policy.o kcf_prov_lib.o kcf_prov_tabs.o \
1530                 kcf_sched.o kcf_session.o kcf_sign.o kcf_spi.o kcf_verify.o \
1531                 kcf_random.o modes.o ecb.o cbc.o ctr.o ccm.o gcm.o \
1532                 fips_random.o

1534 CRYPTOADM_OBJS += cryptoadm.o

1536 CRYPTO_OBJS +=  crypto.o

1538 DPROV_OBJS +=   dprov.o

1540 DCA_OBJS +=     dca.o dca_3des.o dca_debug.o dca_dsa.o dca_kstat.o dca_rng.o \
1541                 dca_rsa.o

1543 AESPROV_OBJS += aes.o aes_impl.o aes_modes.o

1545 ARCFOURPROV_OBJS += arcfour.o arcfour_crypt.o

1547 BLOWFISHPROV_OBJS += blowfish.o blowfish_impl.o

1549 ECCPROV_OBJS += ecc.o ec.o ec2_163.o ec2_mont.o ecdecode.o ecl_mult.o \
1550                 ecp_384.o ecp_jac.o ec2_193.o ecl.o ecp_192.o ecp_521.o \
1551                 ecp_jm.o ec2_233.o ecl_curve.o ecp_224.o ecp_aff.o \
1552                 ecp_mont.o ec2_aff.o ec_naf.o ecl_gf.o ecp_256.o mp_gf2m.o \
1553                 mpi.o mplogic.o mpmontg.o mpprime.o oid.o \
1554                 secitem.o ec2_test.o ecp_test.o

1556 RSAPROV_OBJS += rsa.o rsa_impl.o pkcs1.o

1558 SWRANDPROV_OBJS += swrand.o

1560 #
1561 #                       kernel SSL
1562 #
1563 KSSL_OBJS +=    kssl.o kssl_ioctl.o

1565 KSSL_SOCKFIL_MOD_OBJS += ksslfilter.o ksslapi.o ksslrec.o

1567 #
1568 #                       misc. modules
1569 #

1571 C2AUDIT_OBJS += adr.o audit.o audit_event.o audit_io.o \
1572                 audit_path.o audit_start.o audit_syscalls.o audit_token.o \
1573                 audit_mem.o

1575 PCIC_OBJS +=    pcic.o

1577 RPCSEC_OBJS +=  secmod.o        sec_clnt.o      sec_svc.o       sec_gen.o \
1578                 auth_des.o      auth_kern.o     auth_none.o     auth_loopb.o\
1579                 authdesprt.o    authdesubr.o    authu_prot.o \
```

```
1580                 key_call.o      key_prot.o      svc_authu.o     svcauthdes.o

1582 RPCSEC_GSS_OBJS +=      rpcsec_gssmod.o rpcsec_gss.o rpcsec_gss_misc.o \
1583                 rpcsec_gss_utils.o svc_rpcsec_gss.o

1585 CONSCONFIG_OBJS += consconfig.o

1587 CONSCONFIG_DACF_OBJS  += consconfig_dacf.o consplat.o

1589 TEM_OBJS += tem.o tem_safe.o 6x10.o 7x14.o 12x22.o

1591 KBTRANS_OBJS +=                          \
1592                 kbtrans.o               \
1593                 kbtrans_keytables.o     \
1594                 kbtrans_polled.o        \
1595                 kbtrans_streams.o       \
1596                 usb_keytables.o

1598 KGSSD_OBJS +=   gssd_clnt_stubs.o gssd_handle.o gssd_prot.o \
1599                 gss_display_name.o gss_release_name.o gss_import_name.o \
1600                 gss_release_buffer.o gss_release_oid_set.o gen_oids.o gssdmod.o

1602 KGSSD_DERIVED_OBJS = gssd_xdr.o

1604 KGSS_DUMMY_OBJS += dmech.o

1606 KSOCKET_OBJS += ksocket.o ksocket_mod.o

1608 CRYPTO= cksumtypes.o decrypt.o encrypt.o encrypt_length.o etypes.o \
1609         nfold.o verify_checksum.o prng.o block_size.o make_checksum.o\
1610         checksum_length.o hmac.o default_state.o mandatory_sumtype.o

1612 # crypto/des
1613 CRYPTO_DES= f_cbc.o f_cksum.o f_parity.o weak_key.o d3_cbc.o ef_crypto.o

1615 CRYPTO_DK= checksum.o derive.o dk_decrypt.o dk_encrypt.o

1617 CRYPTO_ARCFOUR= k5_arcfour.o

1619 # crypto/enc_provider
1620 CRYPTO_ENC= des.o des3.o arcfour_provider.o aes_provider.o

1622 # crypto/hash_provider
1623 CRYPTO_HASH= hash_kef_generic.o hash_kmd5.o hash_crc32.o hash_ksha1.o

1625 # crypto/keyhash_provider
1626 CRYPTO_KEYHASH= descbc.o k5_kmd5des.o k_hmac_md5.o

1628 # crypto/crc32
1629 CRYPTO_CRC32= crc32.o

1631 # crypto/old
1632 CRYPTO_OLD= old_decrypt.o old_encrypt.o

1634 # crypto/raw
1635 CRYPTO_RAW= raw_decrypt.o raw_encrypt.o

1637 K5_KRB= kfree.o copy_key.o \
1638         parse.o init_ctx.o \
1639         ser_adata.o ser_addr.o \
1640         ser_auth.o ser_cksum.o \
1641         ser_key.o ser_princ.o \
1642         serialize.o unparse.o \
1643         ser_actx.o

1645 K5_OS=  timeofday.o toffset.o \
```



```
1646         init_os_ctx.o c_ustime.o

1648 SEAL=
1649 # EXPORT DELETE START
1650 SEAL=   seal.o unseal.o
1651 # EXPORT DELETE END

1653 MECH=   delete_sec_context.o \
1654         import_sec_context.o \
1655         gssapi_krb5.o \
1656         k5seal.o k5unseal.o k5sealv3.o \
1657         ser_sctx.o \
1658         sign.o \
1659         util_crypt.o  \
1660         util_validate.o  util_ordering.o  \
1661         util_seqnum.o util_set.o util_seed.o \
1662         wrap_size_limit.o verify.o



1666 MECH_GEN= util_token.o


1669 KGSS_KRB5_OBJS += krb5mech.o \
1670         $(MECH) $(SEAL) $(MECH_GEN) \
1671         $(CRYPTO) $(CRYPTO_DES) $(CRYPTO_DK) $(CRYPTO_ARCFOUR) \
1672         $(CRYPTO_ENC) $(CRYPTO_HASH) \
1673         $(CRYPTO_KEYHASH) $(CRYPTO_CRC32) \
1674         $(CRYPTO_OLD) \
1675         $(CRYPTO_RAW) $(K5_KRB) $(K5_OS)

1677 DES_OBJS +=     des_crypt.o des_impl.o des_ks.o des_soft.o

1679 DLBOOT_OBJS +=  bootparam_xdr.o nfs_dlinet.o scan.o

1681 KRTLD_OBJS +=   kobj_bootflags.o getoptstr.o \
1682                 kobj.o kobj_kdi.o kobj_lm.o kobj_subr.o

1684 MOD_OBJS +=     modctl.o modsubr.o modsysfile.o modconf.o modhash.o

1686 STRPLUMB_OBJS += strplumb.o

1688 CPR_OBJS +=     cpr_driver.o cpr_dump.o \
1689                 cpr_main.o cpr_misc.o cpr_mod.o cpr_stat.o \
1690                 cpr_uthread.o

1692 PROF_OBJS +=    prf.o

1694 SE_OBJS += se_driver.o

1696 SYSACCT_OBJS += acct.o

1698 ACCTCTL_OBJS += acctctl.o

1700 EXACCTSYS_OBJS += exacctsys.o

1702 KAIO_OBJS += aio.o

1704 PCMCIA_OBJS += pcmcia.o cs.o cis.o cis_callout.o cis_handlers.o cis_params.o

1706 BUSRA_OBJS += busra.o

1708 PCS_OBJS += pcs.o

1710 PCAN_OBJS += pcan.o
```

```
1712 PCATA_OBJS += pcide.o pcdisk.o pclabel.o pcata.o

1714 PCSER_OBJS += pcser.o pcser_cis.o

1716 PCWL_OBJS += pcwl.o

1718 PSET_OBJS +=    pset.o

1720 OHCI_OBJS += ohci.o ohci_hub.o ohci_polled.o

1722 UHCI_OBJS += uhci.o uhciutil.o uhcitgt.o uhcihub.o uhcipolled.o

1724 EHCI_OBJS += ehci.o ehci_hub.o ehci_xfer.o ehci_intr.o ehci_util.o ehci_polled.o

1726 HUBD_OBJS += hubd.o

1728 USB_MID_OBJS += usb_mid.o

1730 USB_IA_OBJS += usb_ia.o

1732 UWBA_OBJS += uwba.o uwbai.o

1734 SCSA2USB_OBJS += scsa2usb.o usb_ms_bulkonly.o usb_ms_cbi.o

1736 HWAHC_OBJS += hwahc.o hwahc_util.o

1738 WUSB_DF_OBJS += wusb_df.o
1739 WUSB_FWMOD_OBJS += wusb_fwmod.o

1741 IPF_OBJS += ip_fil_solaris.o fil.o solaris.o ip_state.o ip_frag.o ip_nat.o \
1742             ip_proxy.o ip_auth.o ip_pool.o ip_htable.o ip_lookup.o \
1743             ip_log.o misc.o ip_compat.o ip_nat6.o drand48.o

1745 IBD_OBJS +=     ibd.o ibd_cm.o

1747 EIBNX_OBJS +=   enx_main.o enx_hdlrs.o enx_ibt.o enx_log.o enx_fip.o \
1748                 enx_misc.o enx_q.o enx_ctl.o

1750 EOIB_OBJS +=    eib_adm.o eib_chan.o eib_cmn.o eib_ctl.o eib_data.o \
1751                 eib_fip.o eib_ibt.o eib_log.o eib_mac.o eib_main.o \
1752                 eib_rsrc.o eib_svc.o eib_vnic.o

1754 DLPISTUB_OBJS += dlpistub.o

1756 SDP_OBJS +=     sdpddi.o

1758 TRILL_OBJS +=   trill.o

1760 CTF_OBJS += ctf_create.o ctf_decl.o ctf_error.o ctf_hash.o ctf_labels.o \
1761         ctf_lookup.o ctf_open.o ctf_types.o ctf_util.o ctf_subr.o ctf_mod.o

1763 SMBIOS_OBJS += smb_error.o smb_info.o smb_open.o smb_subr.o smb_dev.o

1765 RPCIB_OBJS += rpcib.o

1767 KMDB_OBJS += kdrv.o

1769 AFE_OBJS += afe.o

1771 BGE_OBJS += bge_main2.o bge_chip2.o bge_kstats.o bge_log.o bge_ndd.o \
1772                 bge_atomic.o bge_mii.o bge_send.o bge_recv2.o bge_mii_5906.o

1774 DMFE_OBJS += dmfe_log.o dmfe_main.o dmfe_mii.o

1776 EFE_OBJS += efe.o
```



```
1778 ELXL_OBJS += elxl.o

1780 HME_OBJS += hme.o

1782 IXGB_OBJS += ixgb.o ixgb_atomic.o ixgb_chip.o ixgb_gld.o ixgb_kstats.o \
1783                 ixgb_log.o ixgb_ndd.o ixgb_rx.o ixgb_tx.o ixgb_xmii.o

1785 NGE_OBJS += nge_main.o nge_atomic.o nge_chip.o nge_ndd.o nge_kstats.o \
1786                 nge_log.o nge_rx.o nge_tx.o nge_xmii.o

1788 PCN_OBJS += pcn.o

1790 RGE_OBJS += rge_main.o rge_chip.o rge_ndd.o rge_kstats.o rge_log.o rge_rxtx.o

1792 URTW_OBJS += urtw.o

1794 ARN_OBJS += arn_hw.o arn_eeprom.o arn_mac.o arn_calib.o arn_ani.o arn_phy.o arn_
1795                 arn_main.o arn_recv.o arn_xmit.o arn_rc.o

1797 ATH_OBJS += ath_aux.o ath_main.o ath_osdep.o ath_rate.o

1799 ATU_OBJS += atu.o

1801 IPW_OBJS += ipw2100_hw.o ipw2100.o

1803 IWI_OBJS += ipw2200_hw.o ipw2200.o

1805 IWH_OBJS += iwh.o

1807 IWK_OBJS += iwk2.o

1809 IWP_OBJS += iwp.o

1811 MWL_OBJS += mwl.o

1813 MWLFW_OBJS += mwlfw_mode.o

1815 WPI_OBJS += wpi.o

1817 RAL_OBJS += rt2560.o ral_rate.o

1819 RUM_OBJS += rum.o

1821 RWD_OBJS += rt2661.o

1823 RWN_OBJS += rt2860.o

1825 UATH_OBJS += uath.o

1827 UATHFW_OBJS += uathfw_mod.o

1829 URAL_OBJS += ural.o

1831 RTW_OBJS += rtw.o smc93cx6.o rtwphy.o rtwphyio.o

1833 ZYD_OBJS += zyd.o zyd_usb.o zyd_hw.o zyd_fw.o

1835 MXFE_OBJS += mxfe.o

1837 MPTSAS_OBJS += mptsas.o mptsas_impl.o mptsas_init.o mptsas_raid.o mptsas_smhba.o

1839 SFE_OBJS += sfe.o sfe_util.o

1841 BFE_OBJS += bfe.o

1843 BRIDGE_OBJS += bridge.o
```

```
1845 IDM_SHARED_OBJS += base64.o

1847 IDM_OBJS +=     $(IDM_SHARED_OBJS) \
1848                 idm.o idm_impl.o idm_text.o idm_conn_sm.o idm_so.o

1850 VR_OBJS += vr.o

1852 ATGE_OBJS += atge_main.o atge_l1e.o atge_mii.o atge_l1.o atge_l1c.o

1854 YGE_OBJS = yge.o

1856 #
1857 #       Build up defines and paths.
1858 #
1859 LINT_DEFS       += -Dunix

1861 #
1862 #       This duality can be removed when the native and target compilers
1863 #       are the same (or at least recognize the same command line syntax!)
1864 #       It is a bug in the current compilation system that the assember
1865 #       can't process the -Y I, flag.
1866 #
1867 NATIVE_INC_PATH += $(INC_PATH) $(CCYFLAG)$(UTSBASE)/common
1868 AS_INC_PATH     += $(INC_PATH) -I$(UTSBASE)/common
1869 INCLUDE_PATH    += $(INC_PATH) $(CCYFLAG)$(UTSBASE)/common

1871 PCIEB_OBJS += pcieb.o

1873 #       Chelsio N110 10G NIC driver module
1874 #
1875 CH_OBJS = ch.o glue.o pe.o sge.o

1877 CH_COM_OBJS =   ch_mac.o ch_subr.o cspi.o espi.o ixf1010.o mc3.o mc4.o mc5.o \
1878                 mv88e1xxx.o mv88x201x.o my3126.o pm3393.o tp.o ulp.o \
1879                 vsc7321.o vsc7326.o xpak.o

1881 #
1882 #       PCI strings file
1883 #
1884 PCI_STRING_OBJS = pci_strings.o

1886 NET_DACF_OBJS += net_dacf.o

1888 #
1889 #       Xframe 10G NIC driver module
1890 #
1891 XGE_OBJS = xge.o xgell.o

1893 XGE_HAL_OBJS =  xgehal-channel.o xgehal-fifo.o xgehal-ring.o  xgehal-config.o \
1894                 xgehal-driver.o  xgehal-mm.o xgehal-stats.o  xgehal-device.o \
1895                 xge-queue.o  xgehal-mgmt.o xgehal-mgmtaux.o

1897 #
1898 #       e1000g module
1899 #
1900 E1000G_OBJS +=  e1000_80003es2lan.o e1000_82540.o e1000_82541.o e1000_82542.o \
1901                 e1000_82543.o e1000_82571.o e1000_api.o e1000_ich8lan.o \
1902                 e1000_mac.o e1000_manage.o e1000_nvm.o e1000_osdep.o \
1903                 e1000_phy.o e1000g_debug.o e1000g_main.o e1000g_alloc.o \
1904                 e1000g_tx.o e1000g_rx.o e1000g_stat.o

1906 #
1907 #       Intel 82575 1G NIC driver module
1908 #
1909 IGB_OBJS =      igb_82575.o igb_api.o igb_mac.o igb_manage.o \
```



```
1910                 igb_nvm.o igb_osdep.o igb_phy.o igb_buf.o \
1911                 igb_debug.o igb_gld.o igb_log.o igb_main.o \
1912                 igb_rx.o igb_stat.o igb_tx.o

1914 #
1915 #       Intel Pro/100 NIC driver module
1916 #
1917 IPRB_OBJS =     iprb.o

1919 #
1920 #       Intel 10GbE PCIE NIC driver module
1921 #
1922 IXGBE_OBJS =    ixgbe_82598.o ixgbe_82599.o ixgbe_api.o         \
1923                 ixgbe_common.o ixgbe_phy.o                      \
1924                 ixgbe_buf.o ixgbe_debug.o ixgbe_gld.o           \
1925                 ixgbe_log.o ixgbe_main.o                        \
1926                 ixgbe_osdep.o ixgbe_rx.o ixgbe_stat.o           \
1927                 ixgbe_tx.o  ixgbe_x540.o ixgbe_mbx.o

1929 #
1930 #       NIU 10G/1G driver module
1931 #
1932 NXGE_OBJS =     nxge_mac.o nxge_ipp.o nxge_rxdma.o              \
1933                 nxge_txdma.o nxge_txc.o nxge_main.o             \
1934                 nxge_hw.o nxge_fzc.o nxge_virtual.o             \
1935                 nxge_send.o nxge_classify.o nxge_fflp.o         \
1936                 nxge_fflp_hash.o nxge_ndd.o nxge_kstats.o       \
1937                 nxge_zcp.o nxge_fm.o nxge_espc.o nxge_hv.o      \
1938                 nxge_hio.o nxge_hio_guest.o nxge_intr.o

1940 NXGE_NPI_OBJS = \
1941                 npi.o npi_mac.o npi_ipp.o                       \
1942                 npi_txdma.o npi_rxdma.o npi_txc.o               \
1943                 npi_zcp.o npi_espc.o npi_fflp.o                 \
1944                 npi_vir.o

1946 NXGE_HCALL_OBJS =       \
1947                 nxge_hcall.o

1949 #
1950 # Virtio modules
1951 #

1953 # Virtio core
1954 VIRTIO_OBJS = virtio.o

1956 # Virtio block driver
1957 VIOBLK_OBJS = vioblk.o

1959 #
1960 #       kiconv modules
1961 #
1962 KICONV_EMEA_OBJS += kiconv_emea.o

1964 KICONV_JA_OBJS += kiconv_ja.o

1966 KICONV_KO_OBJS += kiconv_cck_common.o kiconv_ko.o

1968 KICONV_SC_OBJS += kiconv_cck_common.o kiconv_sc.o

1970 KICONV_TC_OBJS += kiconv_cck_common.o kiconv_tc.o

1972 #
1973 #       AAC module
1974 #
1975 AAC_OBJS = aac.o aac_ioctl.o
```

```
1977 #
1978 #       sdcard modules
1979 #
1980 SDA_OBJS =      sda_cmd.o sda_host.o sda_init.o sda_mem.o sda_mod.o sda_slot.o
1981 SDHOST_OBJS =   sdhost.o

1983 #
1984 #       hxge 10G driver module
1985 #
1986 HXGE_OBJS =     hxge_main.o hxge_vmac.o hxge_send.o              \
1987                 hxge_txdma.o hxge_rxdma.o hxge_virtual.o        \
1988                 hxge_fm.o hxge_fzc.o hxge_hw.o hxge_kstats.o    \
1989                 hxge_ndd.o hxge_pfc.o                           \
1990                 hpi.o hpi_vmac.o hpi_rxdma.o hpi_txdma.o        \
1991                 hpi_vir.o hpi_pfc.o

1993 #
1994 #       MEGARAID_SAS module
1995 #
1996 MEGA_SAS_OBJS = megaraid_sas.o

1998 #
1999 #       MR_SAS module
2000 #
2001 MR_SAS_OBJS = ld_pd_map.o mr_sas.o mr_sas_tbolt.o mr_sas_list.o

2003 #
2004 #       ISCSI_INITIATOR module
2005 #
2006 ISCSI_INITIATOR_OBJS =  chap.o iscsi_io.o iscsi_thread.o        \
2007                         iscsi_ioctl.o iscsid.o iscsi.o          \
2008                         iscsi_login.o isns_client.o iscsiAuthClient.o   \
2009                         iscsi_lun.o iscsiAuthClientGlue.o       \
2010                         iscsi_net.o nvfile.o iscsi_cmd.o        \
2011                         iscsi_queue.o persistent.o iscsi_conn.o \
2012                         iscsi_sess.o radius_auth.o iscsi_crc.o  \
2013                         iscsi_stats.o radius_packet.o iscsi_doorclt.o   \
2014                         iscsi_targetparam.o utils.o kifconf.o

2016 #
2017 #       ntxn 10Gb/1Gb NIC driver module
2018 #
2019 NTXN_OBJS =     unm_nic_init.o unm_gem.o unm_nic_hw.o unm_ndd.o \
2020                         unm_nic_main.o unm_nic_isr.o unm_nic_ctx.o niu.o

2022 #
2023 #       Myricom 10Gb NIC driver module
2024 #
2025 MYRI10GE_OBJS = myri10ge.o myri10ge_lro.o

2027 #       nulldriver module
2028 #
2029 NULLDRIVER_OBJS =       nulldriver.o

2031 TPM_OBJS =      tpm.o tpm_hcall.o
```

****
1546 Tue Jan 15 15:38:23 2013
new/usr/src/uts/common/fs/zfs/THIRDPARTYLICENSE.lz4
%B
****

```
LZ4 - Fast LZ compression algorithm
Copyright (C) 2011-2013, Yann Collet.
BSD 2-Clause License (http://www.opensource.org/licenses/bsd-license.php)

Redistribution and use in source and binary forms, with or without
modification, are permitted provided that the following conditions are
met:

 * Redistributions of source code must retain the above copyright
   notice, this list of conditions and the following disclaimer.

 * Redistributions in binary form must reproduce the above copyright
   notice, this list of conditions and the following disclaimer in the
   documentation and/or other materials provided with the distribution.

THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS
IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED
TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A
PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT OWNER
OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL,
EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO,
PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR
PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF
LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING
NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS
SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

You can contact the author at :
- LZ4 homepage : http://fastcompression.blogspot.com/p/lz4.html
- LZ4 source repository : http://code.google.com/p/lz4/
```

```
**********************************************************
      37 Tue Jan 15 15:38:23 2013
new/usr/src/uts/common/fs/zfs/THIRDPARTYLICENSE.lz4.descrip
%B
**********************************************************
   1 LZ4 COMPRESSION FUNCTIONALITY IN ZFS
```

```
**********************************************************
   30325 Tue Jan 15 15:38:23 2013
new/usr/src/uts/common/fs/zfs/lz4.c
%B
**********************************************************
```

```
/*
 * LZ4 - Fast LZ compression algorithm
 * Header File
 * Copyright (C) 2011-2013, Yann Collet.
 * BSD 2-Clause License (http://www.opensource.org/licenses/bsd-license.php)
 *
 * Redistribution and use in source and binary forms, with or without
 * modification, are permitted provided that the following conditions are
 * met:
 *
 *     * Redistributions of source code must retain the above copyright
 * notice, this list of conditions and the following disclaimer.
 *     * Redistributions in binary form must reproduce the above
 * copyright notice, this list of conditions and the following disclaimer
 * in the documentation and/or other materials provided with the
 * distribution.
 *
 * THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS
 * "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT
 * LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR
 * A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT
 * OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL,
 * SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT
 * LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE,
 * DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY
 * THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT
 * (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE
 * OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.
 *
 * You can contact the author at :
 * - LZ4 homepage : http://fastcompression.blogspot.com/p/lz4.html
 * - LZ4 source repository : http://code.google.com/p/lz4/
 */

#include <sys/zfs_context.h>
#include <sys/lz4hc.h>

static int real_LZ4_compress(const char *source, char *dest, int isize,
    int osize);
static int real_LZ4_uncompress(const char *source, char *dest, int osize);
static int LZ4_compressBound(int isize);
static int LZ4_uncompress_unknownOutputSize(const char *source, char *dest,
    int isize, int maxOutputSize);
static int LZ4_compressCtx(void *ctx, const char *source, char *dest,
    int isize, int osize);
static int LZ4_compress64kCtx(void *ctx, const char *source, char *dest,
    int isize, int osize);

size_t
lz4_compress(void *s_start, void *d_start, size_t s_len, size_t d_len, int n)
{
        uint32_t bufsiz;
        char *dest = d_start;

        ASSERT(d_len >= sizeof (bufsiz));

        /*
         * Switch between high-compression and fast-compression. Decompression
         * is performed by the same routine.
         */
        if (n == 1)
```



```
                bufsiz = zfs_LZ4_compressHC(s_start, &dest[sizeof (bufsiz)],
                    s_len, d_len - sizeof (bufsiz));
        else
                bufsiz = real_LZ4_compress(s_start, &dest[sizeof (bufsiz)],
                    s_len, d_len - sizeof (bufsiz));

        /* Signal an error if the compression routine returned zero. */
        if (bufsiz == 0)
                return (s_len);

        /*
         * Encode the compresed buffer size at the start. We'll need this in
         * decompression to counter the effects of padding which might be
         * added to the compressed buffer and which, if unhandled, would
         * confuse the hell out of our decompression function.
         */
        *(uint32_t *)dest = BE_32(bufsiz);

        return (bufsiz + sizeof (bufsiz));
}

/*ARGSUSED*/
int
lz4_decompress(void *s_start, void *d_start, size_t s_len, size_t d_len, int n)
{
        const char *src = s_start;
        uint32_t bufsiz = BE_IN32(src);

        /* invalid compressed buffer size encoded at start */
        if (bufsiz + sizeof (bufsiz) > s_len)
                return (1);

        /*
         * Returns 0 on success (decompression function returned non-negative)
         * and non-zero on failure (decompression function returned negative.
         */
        return (LZ4_uncompress_unknownOutputSize(&src[sizeof (bufsiz)],
            d_start, bufsiz, d_len) < 0);
}

/*
 * LZ4 API Description:
 *
 * Simple Functions:
 * real_LZ4_compress() :
 *      isize  : is the input size. Max supported value is ~1.9GB
 *      return : the number of bytes written in buffer dest
 *               or 0 if the compression fails (if LZ4_COMPRESSMIN is set).
 *      note : destination buffer must be already allocated.
 *              destination buffer must be sized to handle worst cases
 *              situations (input data not compressible) worst case size
 *              evaluation is provided by function LZ4_compressBound().
 *
 * real_LZ4_uncompress() :
 *      osize  : is the output size, therefore the original size
 *      return : the number of bytes read in the source buffer.
 *              If the source stream is malformed, the function will stop
 *              decoding and return a negative result, indicating the byte
 *              position of the faulty instruction. This function never
 *              writes beyond dest + osize, and is therefore protected
 *              against malicious data packets.
 *      note : destination buffer must be already allocated
 *
 * Advanced Functions
 *
 * LZ4_compressBound() :
```

```
 *      Provides the maximum size that LZ4 may output in a "worst case"
 *      scenario (input data not compressible) primarily useful for memory
 *      allocation of output buffer.
 *
 *      isize  : is the input size. Max supported value is ~1.9GB
 *      return : maximum output size in a "worst case" scenario
 *      note : this function is limited by "int" range (2^31-1)
 *
 * LZ4_uncompress_unknownOutputSize() :
 *      isize  : is the input size, therefore the compressed size
 *      maxOutputSize : is the size of the destination buffer (which must be
 *              already allocated)
 *      return : the number of bytes decoded in the destination buffer
 *              (necessarily <= maxOutputSize). If the source stream is
 *              malformed, the function will stop decoding and return a
 *              negative result, indicating the byte position of the faulty
 *              instruction. This function never writes beyond dest +
 *              maxOutputSize, and is therefore protected against malicious
 *              data packets.
 *      note   : Destination buffer must be already allocated.
 *              This version is slightly slower than real_LZ4_uncompress()
 *
 * LZ4_compressCtx() :
 *      This function explicitly handles the CTX memory structure.
 *
 *      ILLUMOS CHANGES: the CTX memory structure must be explicitly allocated
 *      by the caller (either on the stack or using kmem_zalloc). Passing NULL
 *      isn't valid.
 *
 * LZ4_compress64kCtx() :
 *      Same as LZ4_compressCtx(), but specific to small inputs (<64KB).
 *      isize *Must* be <64KB, otherwise the output will be corrupted.
 *
 *      ILLUMOS CHANGES: the CTX memory structure must be explicitly allocated
 *      by the caller (either on the stack or using kmem_zalloc). Passing NULL
 *      isn't valid.
 */

/*
 * Tuning parameters
 */

/*
 * COMPRESSIONLEVEL: Increasing this value improves compression ratio
 *       Lowering this value reduces memory usage. Reduced memory usage
 *      typically improves speed, due to cache effect (ex: L1 32KB for Intel,
 *      L1 64KB for AMD). Memory usage formula : N->2^(N+2) Bytes
 *      (examples : 12 -> 16KB ; 17 -> 512KB)
 */
#define COMPRESSIONLEVEL 12

/*
 * NOTCOMPRESSIBLE_CONFIRMATION: Decreasing this value will make the
 *      algorithm skip faster data segments considered "incompressible".
 *      This may decrease compression ratio dramatically, but will be
 *      faster on incompressible data. Increasing this value will make
 *      the algorithm search more before declaring a segment "incompressible".
 *      This could improve compression a bit, but will be slower on
 *      incompressible data. The default value (6) is recommended.
 */
#define NOTCOMPRESSIBLE_CONFIRMATION 6

/*
 * BIG_ENDIAN_NATIVE_BUT_INCOMPATIBLE: This will provide a boost to
 * performance for big endian cpu, but the resulting compressed stream
 * will be incompatible with little-endian CPU. You can set this option
```



```
 * to 1 in situations where data will stay within closed environment.
 * This option is useless on Little_Endian CPU (such as x86).
 */
/* #define      BIG_ENDIAN_NATIVE_BUT_INCOMPATIBLE 1 */

/*
 * CPU Feature Detection
 */

/* 32 or 64 bits ? */
#if (defined(__x86_64__) || defined(__x86_64) || defined(__amd64__) || \
    defined(__amd64) || defined(__ppc64__) || defined(_WIN64) || \
    defined(__LP64__) || defined(_LP64))
#define LZ4_ARCH64 1
/*
 * Illumos: On amd64 we have 20k of stack and 24k on sun4u and sun4v, so we
 * can spend 16k on the algorithm
 */
#define STACKLIMIT 12
#else
#define LZ4_ARCH64 0
/*
 * Illumos: On i386 we only have 12k of stack, so in order to maintain the
 * same COMPRESSIONLEVEL we have to use heap allocation. Performance will
 * suck, but alas, it's ZFS on 32-bit we're talking about, so...
 */
#define STACKLIMIT 11
#endif

/*
 * Little Endian or Big Endian?
 * Note: overwrite the below #define if you know your architecture endianess.
 */
#if (defined(__BIG_ENDIAN__) || defined(__BIG_ENDIAN) || \
        defined(_BIG_ENDIAN) || defined(_ARCH_PPC) || defined(__PPC__) || \
        defined(__PPC) || defined(PPC) || defined(__powerpc__) || \
        defined(__powerpc) || defined(powerpc) || \
        ((defined(__BYTE_ORDER__)&&(__BYTE_ORDER__ == __ORDER_BIG_ENDIAN__))))
#define LZ4_BIG_ENDIAN 1
#else
/*
 * Little Endian assumed. PDP Endian and other very rare endian format
 * are unsupported.
 */
#endif

/*
 * Unaligned memory access is automatically enabled for "common" CPU,
 * such as x86. For others CPU, the compiler will be more cautious, and
 * insert extra code to ensure aligned access is respected. If you know
 * your target CPU supports unaligned memory access, you may want to
 * force this option manually to improve performance
 */
#if defined(__ARM_FEATURE_UNALIGNED)
#define LZ4_FORCE_UNALIGNED_ACCESS 1
#endif

/*
 * Illumos: we can't use GCC's __builtin_ctz family of builtins in the
 * kernel
 */
#define LZ4_FORCE_SW_BITCOUNT

/*
 * Compiler Options
 */
```

```
#if __STDC_VERSION__ >= 199901L /* C99 */
/* "restrict" is a known keyword */
#else
/* Disable restrict */
#define restrict
#endif

#define GCC_VERSION (__GNUC__ * 100 + __GNUC_MINOR__)

#ifdef _MSC_VER
/* Visual Studio */
/* Visual is not C99, but supports some kind of inline */
#define inline __forceinline
#if LZ4_ARCH64
/* For Visual 2005 */
#pragma intrinsic(_BitScanForward64)
#pragma intrinsic(_BitScanReverse64)
#else /* !LZ4_ARCH64 */
/* For Visual 2005 */
#pragma intrinsic(_BitScanForward)
#pragma intrinsic(_BitScanReverse)
#endif /* !LZ4_ARCH64 */
#endif /* _MSC_VER */

#ifdef _MSC_VER
#define lz4_bswap16(x) _byteswap_ushort(x)
#else /* !_MSC_VER */
#define lz4_bswap16(x) ((unsigned short int) ((((x) >> 8) & 0xffu) | \
        (((x) & 0xffu) << 8)))
#endif /* !_MSC_VER */

#if (GCC_VERSION >= 302) || (__INTEL_COMPILER >= 800) || defined(__clang__)
#define expect(expr, value)    (__builtin_expect((expr), (value)))
#else
#define expect(expr, value)    (expr)
#endif

#define likely(expr)    expect((expr) != 0, 1)
#define unlikely(expr)  expect((expr) != 0, 0)

/* Basic types */
#if defined(_MSC_VER)
/* Visual Studio does not support 'stdint' natively */
#define BYTE    unsigned __int8
#define U16     unsigned __int16
#define U32     unsigned __int32
#define S32     __int32
#define U64     unsigned __int64
#else /* !defined(_MSC_VER) */
#define BYTE    uint8_t
#define U16     uint16_t
#define U32     uint32_t
#define S32     int32_t
#define U64     uint64_t
#endif /* !defined(_MSC_VER) */

#ifndef LZ4_FORCE_UNALIGNED_ACCESS
#pragma pack(1)
#endif

typedef struct _U16_S {
        U16 v;
} U16_S;
typedef struct _U32_S {
        U32 v;
} U32_S;
```



```
typedef struct _U64_S {
        U64 v;
} U64_S;

#ifndef LZ4_FORCE_UNALIGNED_ACCESS
#pragma pack()
#endif

#define A64(x) (((U64_S *)(x))->v)
#define A32(x) (((U32_S *)(x))->v)
#define A16(x) (((U16_S *)(x))->v)

/*
 * Constants
 */
#define MINMATCH 4

#define HASH_LOG COMPRESSIONLEVEL
#define HASHTABLESIZE (1 << HASH_LOG)
#define HASH_MASK (HASHTABLESIZE - 1)

#define SKIPSTRENGTH (NOTCOMPRESSIBLE_CONFIRMATION > 2 ? \
        NOTCOMPRESSIBLE_CONFIRMATION : 2)

/*
 * Defines if memory is allocated into the stack (local variable),
 * or into the heap (kmem_alloc()).
 */
#define HEAPMODE (HASH_LOG > STACKLIMIT)
#define COPYLENGTH 8
#define LASTLITERALS 5
#define MFLIMIT (COPYLENGTH + MINMATCH)
#define MINLENGTH (MFLIMIT + 1)

#define MAXD_LOG 16
#define MAX_DISTANCE ((1 << MAXD_LOG) - 1)

#define ML_BITS 4
#define ML_MASK ((1U<<ML_BITS)-1)
#define RUN_BITS (8-ML_BITS)
#define RUN_MASK ((1U<<RUN_BITS)-1)


/*
 * Architecture-specific macros
 */
#if LZ4_ARCH64
#define STEPSIZE 8
#define UARCH U64
#define AARCH A64
#define LZ4_COPYSTEP(s, d)      A64(d) = A64(s); d += 8; s += 8;
#define LZ4_COPYPACKET(s, d)    LZ4_COPYSTEP(s, d)
#define LZ4_SECURECOPY(s, d, e) if (d < e) LZ4_WILDCOPY(s, d, e)
#define HTYPE U32
#define INITBASE(base)          const BYTE* const base = ip
#else /* !LZ4_ARCH64 */
#define STEPSIZE 4
#define UARCH U32
#define AARCH A32
#define LZ4_COPYSTEP(s, d)      A32(d) = A32(s); d += 4; s += 4;
#define LZ4_COPYPACKET(s, d)    LZ4_COPYSTEP(s, d); LZ4_COPYSTEP(s, d);
#define LZ4_SECURECOPY          LZ4_WILDCOPY
#define HTYPE const BYTE *
#define INITBASE(base)          const int base = 0
#endif /* !LZ4_ARCH64 */
```

```
#if (defined(LZ4_BIG_ENDIAN) && !defined(BIG_ENDIAN_NATIVE_BUT_INCOMPATIBLE))
#define LZ4_READ_LITTLEENDIAN_16(d, s, p) \
        { U16 v = A16(p); v = lz4_bswap16(v); d = (s) - v; }
#define LZ4_WRITE_LITTLEENDIAN_16(p, i) \
        { U16 v = (U16)(i); v = lz4_bswap16(v); A16(p) = v; p += 2; }
#else
#define LZ4_READ_LITTLEENDIAN_16(d, s, p) { d = (s) - A16(p); }
#define LZ4_WRITE_LITTLEENDIAN_16(p, v)  { A16(p) = v; p += 2; }
#endif


/* Local structures */
struct refTables {
        HTYPE hashTable[HASHTABLESIZE];
};


/* Macros */
#define LZ4_HASH_FUNCTION(i) (((i) * 2654435761U) >> ((MINMATCH * 8) - \
        HASH_LOG))
#define LZ4_HASH_VALUE(p) LZ4_HASH_FUNCTION(A32(p))
#define LZ4_WILDCOPY(s, d, e) do { LZ4_COPYPACKET(s, d) } while (d < e);
#define LZ4_BLINDCOPY(s, d, l) { BYTE* e = (d) + l; LZ4_WILDCOPY(s, d, e); \
        d = e; }


/* Private functions */
#if LZ4_ARCH64

static inline int
LZ4_NbCommonBytes(register U64 val)
{
#if defined(LZ4_BIG_ENDIAN)
#if defined(_MSC_VER) && !defined(LZ4_FORCE_SW_BITCOUNT)
        unsigned long r = 0;
        _BitScanReverse64(&r, val);
        return (int)(r >> 3);
#elif defined(__GNUC__) && (GCC_VERSION >= 304) && \
        !defined(LZ4_FORCE_SW_BITCOUNT)
        return (__builtin_clzll(val) >> 3);
#else
        int r;
        if (!(val >> 32)) {
                r = 4;
        } else {
                r = 0;
                val >>= 32;
        }
        if (!(val >> 16)) {
                r += 2;
                val >>= 8;
        } else {
                val >>= 24;
        }
        r += (!val);
        return (r);
#endif
#else
#if defined(_MSC_VER) && !defined(LZ4_FORCE_SW_BITCOUNT)
        unsigned long r = 0;
        _BitScanForward64(&r, val);
        return (int)(r >> 3);
#elif defined(__GNUC__) && (GCC_VERSION >= 304) && \
        !defined(LZ4_FORCE_SW_BITCOUNT)
        return (__builtin_ctzll(val) >> 3);
#else
```



```
        static const int DeBruijnBytePos[64] =
            { 0, 0, 0, 0, 0, 1, 1, 2, 0, 3, 1, 3, 1, 4, 2, 7, 0, 2, 3, 6, 1, 5,
                3, 5, 1, 3, 4, 4, 2, 5, 6, 7, 7, 0, 1, 2, 3, 3, 4, 6, 2, 6, 5,
                5, 3, 4, 5, 6, 7, 1, 2, 4, 6, 4,
                4, 5, 7, 2, 6, 5, 7, 6, 7, 7
        };
        return DeBruijnBytePos[((U64) ((val & -val) * 0x0218A392CDABBD3F)) >>
            58];
#endif
#endif
}

#else

static inline int
LZ4_NbCommonBytes(register U32 val)
{
#if defined(LZ4_BIG_ENDIAN)
#if defined(_MSC_VER) && !defined(LZ4_FORCE_SW_BITCOUNT)
        unsigned long r = 0;
        _BitScanReverse(&r, val);
        return (int)(r >> 3);
#elif defined(__GNUC__) && (GCC_VERSION >= 304) && \
        !defined(LZ4_FORCE_SW_BITCOUNT)
        return (__builtin_clz(val) >> 3);
#else
        int r;
        if (!(val >> 16)) {
                r = 2;
                val >>= 8;
        } else {
                r = 0;
                val >>= 24;
        }
        r += (!val);
        return (r);
#endif
#else
#if defined(_MSC_VER) && !defined(LZ4_FORCE_SW_BITCOUNT)
        unsigned long r = 0;
        _BitScanForward(&r, val);
        return (int)(r >> 3);
#elif defined(__GNUC__) && (GCC_VERSION >= 304) && \
        !defined(LZ4_FORCE_SW_BITCOUNT)
        return (__builtin_ctz(val) >> 3);
#else
        static const int DeBruijnBytePos[32] = {
                0, 0, 3, 0, 3, 1, 3, 0,
                3, 2, 2, 1, 3, 2, 0, 1,
                3, 3, 1, 2, 2, 2, 2, 0,
                3, 1, 2, 0, 1, 0, 1, 1
        };
        return DeBruijnBytePos[((U32) ((val & -(S32) val) * 0x077CB531U)) >>
            27];
#endif
#endif
}

#endif

/* Public functions */

static int
LZ4_compressBound(int isize)
{
        return (isize + (isize / 255) + 16);
```

```
524 }

526 /* Compression functions */

528 /*ARGSUSED*/
529 static int
530 LZ4_compressCtx(void *ctx, const char *source, char *dest, int isize,
531     int osize)
532 {
533 #if HEAPMODE
534         struct refTables *srt = (struct refTables *)ctx;
535         HTYPE *HashTable = (HTYPE *) (srt->hashTable);
536 #else
537         HTYPE HashTable[HASHTABLESIZE] = { 0 };
538 #endif

540         const BYTE *ip = (BYTE *) source;
541         INITBASE(base);
542         const BYTE *anchor = ip;
543         const BYTE *const iend = ip + isize;
544         const BYTE *const oend = (BYTE *) dest + osize;
545         const BYTE *const mflimit = iend - MFLIMIT;
546 #define matchlimit (iend - LASTLITERALS)

548         BYTE *op = (BYTE *) dest;

550         int len, length;
551         const int skipStrength = SKIPSTRENGTH;
552         U32 forwardH;


555         /* Init */
556         if (isize < MINLENGTH)
557                 goto _last_literals;

559         /* First Byte */
560         HashTable[LZ4_HASH_VALUE(ip)] = ip - base;
561         ip++;
562         forwardH = LZ4_HASH_VALUE(ip);

564         /* Main Loop */
565         for (;;) {
566                 int findMatchAttempts = (1U << skipStrength) + 3;
567                 const BYTE *forwardIp = ip;
568                 const BYTE *ref;
569                 BYTE *token;

571                 /* Find a match */
572                 do {
573                         U32 h = forwardH;
574                         int step = findMatchAttempts++ >> skipStrength;
575                         ip = forwardIp;
576                         forwardIp = ip + step;

578                         if unlikely(forwardIp > mflimit) {
579                                 goto _last_literals;
580                         }

582                         forwardH = LZ4_HASH_VALUE(forwardIp);
583                         ref = base + HashTable[h];
584                         HashTable[h] = ip - base;

586                 } while ((ref < ip - MAX_DISTANCE) || (A32(ref) != A32(ip)));

588                 /* Catch up */
589                 while ((ip > anchor) && (ref > (BYTE *) source) &&
```



```
590                     unlikely(ip[-1] == ref[-1])) {
591                         ip--;
592                         ref--;
593                 }

595                 /* Encode Literal length */
596                 length = ip - anchor;
597                 token = op++;

599                 /* Check output limit */
600                 if unlikely(op + length + (2 + 1 + LASTLITERALS) +
601                     (length >> 8) > oend)
602                         return (0);

604                 if (length >= (int)RUN_MASK) {
605                         *token = (RUN_MASK << ML_BITS);
606                         len = length - RUN_MASK;
607                         for (; len > 254; len -= 255)
608                                 *op++ = 255;
609                         *op++ = (BYTE)len;
610                 } else
611                         *token = (length << ML_BITS);

613                 /* Copy Literals */
614                 LZ4_BLINDCOPY(anchor, op, length);

616                 _next_match:
617                 /* Encode Offset */
618                 LZ4_WRITE_LITTLEENDIAN_16(op, ip - ref);

620                 /* Start Counting */
621                 ip += MINMATCH;
622                 ref += MINMATCH;        /* MinMatch verified */
623                 anchor = ip;
624                 while likely(ip < matchlimit - (STEPSIZE - 1)) {
625                         UARCH diff = AARCH(ref) ^ AARCH(ip);
626                         if (!diff) {
627                                 ip += STEPSIZE;
628                                 ref += STEPSIZE;
629                                 continue;
630                         }
631                         ip += LZ4_NbCommonBytes(diff);
632                         goto _endCount;
633                 }
634 #if LZ4_ARCH64
635                 if ((ip < (matchlimit - 3)) && (A32(ref) == A32(ip))) {
636                         ip += 4;
637                         ref += 4;
638                 }
639 #endif
640                 if ((ip < (matchlimit - 1)) && (A16(ref) == A16(ip))) {
641                         ip += 2;
642                         ref += 2;
643                 }
644                 if ((ip < matchlimit) && (*ref == *ip))
645                         ip++;
646                 _endCount:

648                 /* Encode MatchLength */
649                 len = (ip - anchor);
650                 /* Check output limit */
651                 if unlikely(op + (1 + LASTLITERALS) + (len >> 8) > oend)
652                         return (0);
653                 if (len >= (int)ML_MASK) {
654                         *token += ML_MASK;
655                         len -= ML_MASK;
```

```
656                         for (; len > 509; len -= 510) {
657                                 *op++ = 255;
658                                 *op++ = 255;
659                         }
660                         if (len > 254) {
661                                 len -= 255;
662                                 *op++ = 255;
663                         }
664                         *op++ = (BYTE)len;
665                 } else
666                         *token += len;
667
668                 /* Test end of chunk */
669                 if (ip > mflimit) {
670                         anchor = ip;
671                         break;
672                 }
673                 /* Fill table */
674                 HashTable[LZ4_HASH_VALUE(ip - 2)] = ip - 2 - base;
675
676                 /* Test next position */
677                 ref = base + HashTable[LZ4_HASH_VALUE(ip)];
678                 HashTable[LZ4_HASH_VALUE(ip)] = ip - base;
679                 if ((ref > ip - (MAX_DISTANCE + 1)) && (A32(ref) == A32(ip))) {
680                         token = op++;
681                         *token = 0;
682                         goto _next_match;
683                 }
684                 /* Prepare next loop */
685                 anchor = ip++;
686                 forwardH = LZ4_HASH_VALUE(ip);
687         }
688
689         _last_literals:
690         /* Encode Last Literals */
691         {
692                 int lastRun = iend - anchor;
693                 if (op + lastRun + 1 + ((lastRun + 255 - RUN_MASK) / 255) >
694                     oend)
695                         return (0);
696                 if (lastRun >= (int)RUN_MASK) {
697                         *op++ = (RUN_MASK << ML_BITS);
698                         lastRun -= RUN_MASK;
699                         for (; lastRun > 254; lastRun -= 255) {
700                                 *op++ = 255;
701                         }
702                         *op++ = (BYTE)lastRun;
703                 } else
704                         *op++ = (lastRun << ML_BITS);
705                 (void) memcpy(op, anchor, iend - anchor);
706                 op += iend - anchor;
707         }
708
709         /* End */
710         return (int)(((char *)op) - dest);
711 }
712
713
714
715 /* Note : this function is valid only if isize < LZ4_64KLIMIT */
716 #define LZ4_64KLIMIT ((1 << 16) + (MFLIMIT - 1))
717 #define HASHLOG64K (HASH_LOG + 1)
718 #define HASH64KTABLESIZE (1U << HASHLOG64K)
719 #define LZ4_HASH64K_FUNCTION(i) (((i) * 2654435761U) >> ((MINMATCH*8) - \
720         HASHLOG64K))
721 #define LZ4_HASH64K_VALUE(p)    LZ4_HASH64K_FUNCTION(A32(p))
```



```
722
723 /*ARGSUSED*/
724 static int
725 LZ4_compress64kCtx(void *ctx, const char *source, char *dest, int isize,
726     int osize)
727 {
728 #if HEAPMODE
729         struct refTables *srt = (struct refTables *)ctx;
730         U16 *HashTable = (U16 *) (srt->hashTable);
731 #else
732         U16 HashTable[HASH64KTABLESIZE] = { 0 };
733 #endif
734
735         const BYTE *ip = (BYTE *) source;
736         const BYTE *anchor = ip;
737         const BYTE *const base = ip;
738         const BYTE *const iend = ip + isize;
739         const BYTE *const oend = (BYTE *) dest + osize;
740         const BYTE *const mflimit = iend - MFLIMIT;
741 #define matchlimit (iend - LASTLITERALS)
742
743         BYTE *op = (BYTE *) dest;
744
745         int len, length;
746         const int skipStrength = SKIPSTRENGTH;
747         U32 forwardH;
748
749         /* Init */
750         if (isize < MINLENGTH)
751                 goto _last_literals;
752
753         /* First Byte */
754         ip++;
755         forwardH = LZ4_HASH64K_VALUE(ip);
756
757         /* Main Loop */
758         for (;;) {
759                 int findMatchAttempts = (1U << skipStrength) + 3;
760                 const BYTE *forwardIp = ip;
761                 const BYTE *ref;
762                 BYTE *token;
763
764                 /* Find a match */
765                 do {
766                         U32 h = forwardH;
767                         int step = findMatchAttempts++ >> skipStrength;
768                         ip = forwardIp;
769                         forwardIp = ip + step;
770
771                         if (forwardIp > mflimit) {
772                                 goto _last_literals;
773                         }
774
775                         forwardH = LZ4_HASH64K_VALUE(forwardIp);
776                         ref = base + HashTable[h];
777                         HashTable[h] = ip - base;
778
779                 } while (A32(ref) != A32(ip));
780
781                 /* Catch up */
782                 while ((ip > anchor) && (ref > (BYTE *) source) &&
783                     (ip[-1] == ref[-1])) {
784                         ip--;
785                         ref--;
786                 }
```

```
788                 /* Encode Literal length */
789                 length = ip - anchor;
790                 token = op++;

792                 /* Check output limit */
793                 if unlikely(op + length + (2 + 1 + LASTLITERALS) +
794                     (length >> 8) > oend)
795                         return (0);

797                 if (length >= (int)RUN_MASK) {
798                         *token = (RUN_MASK << ML_BITS);
799                         len = length - RUN_MASK;
800                         for (; len > 254; len -= 255)
801                                 *op++ = 255;
802                         *op++ = (BYTE)len;
803                 } else
804                         *token = (length << ML_BITS);

806                 /* Copy Literals */
807                 LZ4_BLINDCOPY(anchor, op, length);

809                 _next_match:
810                 /* Encode Offset */
811                 LZ4_WRITE_LITTLEENDIAN_16(op, ip - ref);

813                 /* Start Counting */
814                 ip += MINMATCH;
815                 ref += MINMATCH;        /* MinMatch verified */
816                 anchor = ip;
817                 while (ip < matchlimit - (STEPSIZE - 1)) {
818                         UARCH diff = AARCH(ref) ^ AARCH(ip);
819                         if (!diff) {
820                                 ip += STEPSIZE;
821                                 ref += STEPSIZE;
822                                 continue;
823                         }
824                         ip += LZ4_NbCommonBytes(diff);
825                         goto _endCount;
826                 }
827 #if LZ4_ARCH64
828                 if ((ip < (matchlimit - 3)) && (A32(ref) == A32(ip))) {
829                         ip += 4;
830                         ref += 4;
831                 }
832 #endif
833                 if ((ip < (matchlimit - 1)) && (A16(ref) == A16(ip))) {
834                         ip += 2;
835                         ref += 2;
836                 }
837                 if ((ip < matchlimit) && (*ref == *ip))
838                         ip++;
839                 _endCount:

841                 /* Encode MatchLength */
842                 len = (ip - anchor);
843                 /* Check output limit */
844                 if unlikely(op + (1 + LASTLITERALS) + (len >> 8) > oend)
845                         return (0);
846                 if (len >= (int)ML_MASK) {
847                         *token += ML_MASK;
848                         len -= ML_MASK;
849                         for (; len > 509; len -= 510) {
850                                 *op++ = 255;
851                                 *op++ = 255;
852                         }
853                         if (len > 254) {
```



```
854                                 len -= 255;
855                                 *op++ = 255;
856                         }
857                         *op++ = (BYTE)len;
858                 } else
859                         *token += len;

861                 /* Test end of chunk */
862                 if (ip > mflimit) {
863                         anchor = ip;
864                         break;
865                 }
866                 /* Fill table */
867                 HashTable[LZ4_HASH64K_VALUE(ip - 2)] = ip - 2 - base;

869                 /* Test next position */
870                 ref = base + HashTable[LZ4_HASH64K_VALUE(ip)];
871                 HashTable[LZ4_HASH64K_VALUE(ip)] = ip - base;
872                 if (A32(ref) == A32(ip)) {
873                         token = op++;
874                         *token = 0;
875                         goto _next_match;
876                 }
877                 /* Prepare next loop */
878                 anchor = ip++;
879                 forwardH = LZ4_HASH64K_VALUE(ip);
880         }

882         _last_literals:
883         /* Encode Last Literals */
884         {
885                 int lastRun = iend - anchor;
886                 if (op + lastRun + 1 + ((lastRun + 255 - RUN_MASK) / 255) >
887                     oend)
888                         return (0);
889                 if (lastRun >= (int)RUN_MASK) {
890                         *op++ = (RUN_MASK << ML_BITS);
891                         lastRun -= RUN_MASK;
892                         for (; lastRun > 254; lastRun -= 255)
893                                 *op++ = 255;
894                         *op++ = (BYTE)lastRun;
895                 } else
896                         *op++ = (lastRun << ML_BITS);
897                 (void) memcpy(op, anchor, iend - anchor);
898                 op += iend - anchor;
899         }

901         /* End */
902         return (int)(((char *)op) - dest);
903 }

905 static int
906 real_LZ4_compress(const char *source, char *dest, int isize, int osize)
907 {
908 #if HEAPMODE
909         void *ctx = kmem_zalloc(sizeof (struct refTables), KM_NOSLEEP);
910         int result;

912         /*
913          * out of kernel memory, gently fall through - this will disable
914          * compression in zio_compress_data
915          */
916         if (ctx == NULL)
917                 return (0);

919         if (isize < LZ4_64KLIMIT)
```

```
920                 result = LZ4_compress64kCtx(ctx, source, dest, isize, osize);
921         else
922                 result = LZ4_compressCtx(ctx, source, dest, isize, osize);

924         kmem_free(ctx, sizeof (struct refTables));
925         return (result);
926 #else
927         if (isize < (int)LZ4_64KLIMIT)
928                 return (LZ4_compress64kCtx(NULL, source, dest, isize, osize));
929         return (LZ4_compressCtx(NULL, source, dest, isize, osize));
930 #endif
931 }

933 /* Decompression functions */

935 /*
936  * Note: The decoding functions real_LZ4_uncompress() and
937  *      LZ4_uncompress_unknownOutputSize() are safe against "buffer overflow"
938  *      attack type. They will never write nor read outside of the provided
939  *      output buffers. LZ4_uncompress_unknownOutputSize() also insures that
940  *      it will never read outside of the input buffer. A corrupted input
941  *      will produce an error result, a negative int, indicating the position
942  *      of the error within input stream.
943  */

945 static int
946 real_LZ4_uncompress(const char *source, char *dest, int osize)
947 {
948         /* Local Variables */
949         const BYTE *restrict ip = (const BYTE *) source;
950         const BYTE *ref;

952         BYTE *op = (BYTE *) dest;
953         BYTE *const oend = op + osize;
954         BYTE *cpy;

956         unsigned token;

958         size_t length;
959         size_t dec32table[] = {0, 3, 2, 3, 0, 0, 0, 0};
960 #if LZ4_ARCH64
961         size_t dec64table[] = {0, 0, 0, (size_t)-1, 0, 1, 2, 3};
962 #endif

964         /* Main Loop */
965         for (;;) {
966                 /* get runlength */
967                 token = *ip++;
968                 if ((length = (token >> ML_BITS)) == RUN_MASK) {
969                         size_t len;
970                         for (; (len = *ip++) == 255; length += 255) {
971                         }
972                         length += len;
973                 }
974                 /* copy literals */
975                 cpy = op + length;
976                 if unlikely(cpy > oend - COPYLENGTH) {
977                         if (cpy != oend)
978                                 /* Error: we must necessarily stand at EOF */
979                                 goto _output_error;
980                         (void) memcpy(op, ip, length);
981                         ip += length;
982                         break;  /* EOF */
983                         }
984                 LZ4_WILDCOPY(ip, op, cpy);
985                 ip -= (op - cpy);
```



```
986                 op = cpy;

988                 /* get offset */
989                 LZ4_READ_LITTLEENDIAN_16(ref, cpy, ip);
990                 ip += 2;
991                 if unlikely(ref < (BYTE * const) dest)
992                         /*
993                          * Error: offset create reference outside destination
994                          * buffer
995                          */
996                         goto _output_error;

998                 /* get matchlength */
999                 if ((length = (token & ML_MASK)) == ML_MASK) {
1000                        for (; *ip == 255; length += 255) {
1001                                ip++;
1002                        }
1003                        length += *ip++;
1004                }
1005                /* copy repeated sequence */
1006                if unlikely(op - ref < STEPSIZE) {
1007 #if LZ4_ARCH64
1008                        size_t dec64 = dec64table[op-ref];
1009 #else
1010                        const int dec64 = 0;
1011 #endif
1012                        op[0] = ref[0];
1013                        op[1] = ref[1];
1014                        op[2] = ref[2];
1015                        op[3] = ref[3];
1016                        op += 4;
1017                        ref += 4;
1018                        ref -= dec32table[op-ref];
1019                        A32(op) = A32(ref);
1020                        op += STEPSIZE - 4;
1021                        ref -= dec64;
1022                } else {
1023                        LZ4_COPYSTEP(ref, op);
1024                }
1025                cpy = op + length - (STEPSIZE - 4);
1026                if (cpy > oend - COPYLENGTH) {
1027                        if (cpy > oend)
1028                                /*
1029                                 * Error: request to write beyond destination
1030                                 * buffer
1031                                 */
1032                                goto _output_error;
1033                        LZ4_SECURECOPY(ref, op, (oend - COPYLENGTH));
1034                        while (op < cpy)
1035                                *op++ = *ref++;
1036                        op = cpy;
1037                        if (op == oend)
1038                                /*
1039                                 * Check EOF (should never happen, since last
1040                                 * 5 bytes are supposed to be literals)
1041                                 */
1042                                goto _output_error;
1043                        continue;
1044                }
1045                LZ4_SECURECOPY(ref, op, cpy);
1046                op = cpy;       /* correction */
1047        }

1049        /* end of decoding */
1050        return (int)(((char *)ip) - source);
```

```
1052         /* write overflow error detected */
1053         _output_error:
1054         return (int)(-(((char *)ip) - source));
1055 }

1057 static int
1058 LZ4_uncompress_unknownOutputSize(const char *source, char *dest, int isize,
1059     int maxOutputSize)
1060 {
1061         /* Local Variables */
1062         const BYTE *restrict ip = (const BYTE *) source;
1063         const BYTE *const iend = ip + isize;
1064         const BYTE *ref;

1066         BYTE *op = (BYTE *) dest;
1067         BYTE *const oend = op + maxOutputSize;
1068         BYTE *cpy;

1070         size_t dec32table[] = {0, 3, 2, 3, 0, 0, 0, 0};
1071 #if LZ4_ARCH64
1072         size_t dec64table[] = {0, 0, 0, (size_t)-1, 0, 1, 2, 3};
1073 #endif

1075         /* Main Loop */
1076         while (ip < iend) {
1077                 unsigned token;
1078                 size_t length;

1080                 /* get runlength */
1081                 token = *ip++;
1082                 if ((length = (token >> ML_BITS)) == RUN_MASK) {
1083                         int s = 255;
1084                         while ((ip < iend) && (s == 255)) {
1085                                 s = *ip++;
1086                                 length += s;
1087                         }
1088                 }
1089                 /* copy literals */
1090                 cpy = op + length;
1091                 if ((cpy > oend - COPYLENGTH) ||
1092                     (ip + length > iend - COPYLENGTH)) {
1093                         if (cpy > oend)
1094                                 /* Error: writes beyond output buffer */
1095                                 goto _output_error;
1096                         if (ip + length != iend)
1097                                 /*
1098                                  * Error: LZ4 format requires to consume all
1099                                  * input at this stage
1100                                  */
1101                                 goto _output_error;
1102                         (void) memcpy(op, ip, length);
1103                         op += length;
1104                         /* Necessarily EOF, due to parsing restrictions */
1105                         break;
1106                 }
1107                 LZ4_WILDCOPY(ip, op, cpy);
1108                 ip -= (op - cpy);
1109                 op = cpy;

1111                 /* get offset */
1112                 LZ4_READ_LITTLEENDIAN_16(ref, cpy, ip);
1113                 ip += 2;
1114                 if (ref < (BYTE * const) dest)
1115                         /*
1116                          * Error: offset creates reference outside of
1117                          * destination buffer
```



```
1118                          */
1119                         goto _output_error;

1121                 /* get matchlength */
1122                 if ((length = (token & ML_MASK)) == ML_MASK) {
1123                         while (ip < iend) {
1124                                 int s = *ip++;
1125                                 length += s;
1126                                 if (s == 255)
1127                                         continue;
1128                                 break;
1129                         }
1130                 }
1131                 /* copy repeated sequence */
1132                 if unlikely(op - ref < STEPSIZE) {
1133 #if LZ4_ARCH64
1134                         size_t dec64 = dec64table[op-ref];
1135 #else
1136                         const int dec64 = 0;
1137 #endif
1138                         op[0] = ref[0];
1139                         op[1] = ref[1];
1140                         op[2] = ref[2];
1141                         op[3] = ref[3];
1142                         op += 4;
1143                         ref += 4;
1144                         ref -= dec32table[op-ref];
1145                         A32(op) = A32(ref);
1146                         op += STEPSIZE - 4;
1147                         ref -= dec64;
1148                 } else {
1149                         LZ4_COPYSTEP(ref, op);
1150                 }
1151                 cpy = op + length - (STEPSIZE - 4);
1152                 if (cpy > oend - COPYLENGTH) {
1153                         if (cpy > oend)
1154                                 /*
1155                                  * Error: request to write outside of
1156                                  * destination buffer
1157                                  */
1158                                 goto _output_error;
1159                         LZ4_SECURECOPY(ref, op, (oend - COPYLENGTH));
1160                         while (op < cpy)
1161                                 *op++ = *ref++;
1162                         op = cpy;
1163                         if (op == oend)
1164                                 /*
1165                                  * Check EOF (should never happen, since
1166                                  * last 5 bytes are supposed to be literals)
1167                                  */
1168                                 goto _output_error;
1169                         continue;
1170                 }
1171                 LZ4_SECURECOPY(ref, op, cpy);
1172                 op = cpy;       /* correction */
1173         }

1175         /* end of decoding */
1176         return (int)(((char *)op) - dest);

1178         /* write overflow error detected */
1179         _output_error:
1180         return (int)(-(((char *)ip) - source));
1181 }
```

```
**********************************************************
   19792 Tue Jan 15 15:38:24 2013
new/usr/src/uts/common/fs/zfs/lz4hc.c
%B
**********************************************************
```

```
/*
 * LZ4 HC - High Compression Mode of LZ4
 * Copyright (C) 2011-2012, Yann Collet.
 * BSD 2-Clause License (http://www.opensource.org/licenses/bsd-license.php)
 *
 * Redistribution and use in source and binary forms, with or without
 * modification, are permitted provided that the following conditions are
 * met:
 *
 *     * Redistributions of source code must retain the above copyright
 * notice, this list of conditions and the following disclaimer.
 *     * Redistributions in binary form must reproduce the above
 * copyright notice, this list of conditions and the following disclaimer
 * in the documentation and/or other materials provided with the
 * distribution.
 *
 * THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS
 * "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT
 * LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR
 * A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT
 * OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL,
 * SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT
 * LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE,
 * DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY
 * THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT
 * (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE
 * OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.
 *
 * You can contact the author at :
 * - LZ4 homepage : http://fastcompression.blogspot.com/p/lz4.html
 * - LZ4 source repository : http://code.google.com/p/lz4/
 */


/*
 * CPU Feature Detection
 */

/* 32 or 64 bits ? */
#if (defined(__x86_64__) || defined(__x86_64) || defined(__amd64__) || \
    defined(__amd64) || defined(__ppc64__) || defined(_WIN64) || \
    defined(__LP64__) || defined(_LP64))
#define LZ4_ARCH64 1
#else
#define LZ4_ARCH64 0
#endif

/*
 * Little Endian or Big Endian?
 * Note: overwrite the below #define if you know your architecture endianess.
 */
#if (defined(__BIG_ENDIAN__) || defined(__BIG_ENDIAN) || \
        defined(_BIG_ENDIAN) || defined(_ARCH_PPC) || defined(__PPC__) || \
        defined(__PPC) || defined(PPC) || defined(__powerpc__) || \
        defined(__powerpc) || defined(powerpc) || \
        ((defined(__BYTE_ORDER__)&&(__BYTE_ORDER__ == __ORDER_BIG_ENDIAN__))))
#define LZ4_BIG_ENDIAN 1
#else
/*
 * Little Endian assumed. PDP Endian and other very rare endian format
 * are unsupported.
```



```
 */
#endif

/*
 * Unaligned memory access is automatically enabled for "common" CPU,
 * such as x86. For others CPU, the compiler will be more cautious, and
 * insert extra code to ensure aligned access is respected. If you know
 * your target CPU supports unaligned memory access, you may want to
 * force this option manually to improve performance
 */
#if defined(__ARM_FEATURE_UNALIGNED)
#define LZ4_FORCE_UNALIGNED_ACCESS 1
#endif

/*
 * Illumos: we can't use GCC's __builtin_ctz family of builtins in the
 * kernel
 */
#define LZ4_FORCE_SW_BITCOUNT

/*
 * Compiler Options
 */
#if __STDC_VERSION__ >= 199901L /* C99 */
/* "restrict" is a known keyword */
#else
/* Disable restrict */
#define restrict
#endif

#define GCC_VERSION (__GNUC__ * 100 + __GNUC_MINOR__)

#ifdef _MSC_VER
#define lz4_bswap16(x) _byteswap_ushort(x)
#else /* !_MSC_VER */
#define lz4_bswap16(x) ((unsigned short int) ((((x) >> 8) & 0xffu) | \
        (((x) & 0xffu) << 8)))
#endif /* !_MSC_VER */

#if (GCC_VERSION >= 302) || (__INTEL_COMPILER >= 800) || defined(__clang__)
#define expect(expr, value)     (__builtin_expect((expr), (value)))
#else
#define expect(expr, value)     (expr)
#endif

#define likely(expr)    expect((expr) != 0, 1)
#define unlikely(expr)  expect((expr) != 0, 0)

/*
 * Includes
 */
#include <sys/zfs_context.h>

#define ALLOCATOR(s) kmem_alloc(s, KM_NOSLEEP)
#define FREEMEM kmem_free
#define MEM_INIT (void) memset

/* Basic Types */
#if defined(_MSC_VER)
/* Visual Studio does not support 'stdint' natively */
#define BYTE    unsigned __int8
#define U16     unsigned __int16
#define U32     unsigned __int32
#define S32     __int32
#define U64     unsigned __int64
#else /* !defined(_MSC_VER) */
```

```
#define BYTE    uint8_t
#define U16     uint16_t
#define U32     uint32_t
#define S32     int32_t
#define U64     uint64_t
#endif /* !defined(_MSC_VER) */

#ifndef LZ4_FORCE_UNALIGNED_ACCESS
#pragma pack(1)
#endif

typedef struct _U16_S {
        U16 v;
} U16_S;
typedef struct _U32_S {
        U32 v;
} U32_S;
typedef struct _U64_S {
        U64 v;
} U64_S;

#ifndef LZ4_FORCE_UNALIGNED_ACCESS
#pragma pack()
#endif

#define A64(x) (((U64_S *)(x))->v)
#define A32(x) (((U32_S *)(x))->v)
#define A16(x) (((U16_S *)(x))->v)


/* Constants */
#define MINMATCH 4

#define DICTIONARY_LOGSIZE 16
#define MAXD (1<<DICTIONARY_LOGSIZE)
#define MAXD_MASK ((U32)(MAXD - 1))
#define MAX_DISTANCE (MAXD - 1)

#define HASH_LOG (DICTIONARY_LOGSIZE - 1)
#define HASHTABLESIZE (1 << HASH_LOG)
#define HASH_MASK (HASHTABLESIZE - 1)

#define MAX_NB_ATTEMPTS 256

#define ML_BITS  4
#define ML_MASK  (size_t)((1U << ML_BITS) - 1)
#define RUN_BITS (8 - ML_BITS)
#define RUN_MASK ((1U << RUN_BITS) - 1)

#define COPYLENGTH 8
#define LASTLITERALS 5
#define MFLIMIT (COPYLENGTH + MINMATCH)
#define MINLENGTH (MFLIMIT + 1)
#define OPTIMAL_ML (int)((ML_MASK - 1) + MINMATCH)


/* Architecture-specific macros */
#if LZ4_ARCH64
#define STEPSIZE 8
#define LZ4_COPYSTEP(s, d)      A64(d) = A64(s); d += 8; s += 8;
#define LZ4_COPYPACKET(s, d)    LZ4_COPYSTEP(s, d)
#define UARCH U64
#define AARCH A64
#define HTYPE                   U32
#define INITBASE(b, s)          const BYTE* const b = s
#else /* !LZ4_ARCH64 */
```



```
#define STEPSIZE 4
#define LZ4_COPYSTEP(s, d)      A32(d) = A32(s); d += 4; s += 4;
#define LZ4_COPYPACKET(s, d)    LZ4_COPYSTEP(s, d); LZ4_COPYSTEP(s, d);
#define UARCH U32
#define AARCH A32
#define HTYPE                   const BYTE*
#define INITBASE(b, s)          const int b = 0
#endif /* !LZ4_ARCH64 */

#if defined(LZ4_BIG_ENDIAN)
#define LZ4_READ_LITTLEENDIAN_16(d, s, p) \
        { U16 v = A16(p); v = lz4_bswap16(v); d = (s) - v; }
#define LZ4_WRITE_LITTLEENDIAN_16(p, i) \
        { U16 v = (U16)(i); v = lz4_bswap16(v); A16(p) = v; p += 2; }
#else /* !defined(LZ4_BIG_ENDIAN) */
#define LZ4_READ_LITTLEENDIAN_16(d, s, p) \
        { d = (s) - A16(p); }
#define LZ4_WRITE_LITTLEENDIAN_16(p, v) \
        { A16(p) = v; p += 2; }
#endif /* !defined(LZ4_BIG_ENDIAN) */


/* Local Types */
typedef struct {
        const BYTE* base;
        HTYPE hashTable[HASHTABLESIZE];
        U16 chainTable[MAXD];
        const BYTE* nextToUpdate;
} LZ4HC_Data_Structure;


/* Macros */
#define LZ4_WILDCOPY(s, d, e)   do { LZ4_COPYPACKET(s, d) } while (d < e);
#define LZ4_BLINDCOPY(s, d, l)  { BYTE* e = d + l; LZ4_WILDCOPY(s, d, e); \
        d = e; }
#define HASH_FUNCTION(i)        (((i) * 2654435761U) >> ((MINMATCH * 8) - \
        HASH_LOG))
#define HASH_VALUE(p)           HASH_FUNCTION(A32(p))
#define HASH_POINTER(p)         (HashTable[HASH_VALUE(p)] + base)
#define DELTANEXT(p)            chainTable[(size_t)(p) & MAXD_MASK]
#define GETNEXT(p)              ((p) - (size_t)DELTANEXT(p))
#define ADD_HASH(p) \
        { \
                size_t delta = (p) - HASH_POINTER(p); \
                if (delta > MAX_DISTANCE) \
                        delta = MAX_DISTANCE; \
                DELTANEXT(p) = (U16)delta; \
                HashTable[HASH_VALUE(p)] = (p) - base; \
        }


/* Private functions */
#if LZ4_ARCH64

static inline int
LZ4_NbCommonBytes(register U64 val)
{
#if defined(LZ4_BIG_ENDIAN)
#if defined(_MSC_VER) && !defined(LZ4_FORCE_SW_BITCOUNT)
        unsigned long r = 0;
        _BitScanReverse64(&r, val);
        return (int)(r >> 3);
#elif defined(__GNUC__) && ((__GNUC__ * 100 + __GNUC_MINOR__) >= 304) && \
        !defined(LZ4_FORCE_SW_BITCOUNT)
        return (__builtin_clzll(val) >> 3);
#else
```

```
260         int r;
261         if (!(val >> 32))
262                 r = 4;
263         else {
264                 r = 0;
265                 val >>= 32;
266         }
267         if (!(val >> 16)) {
268                 r += 2;
269                 val >>= 8;
270         } else
271                 val >>= 24;
272         r += (!val);
273         return (r);
274 #endif
275 #else /* !defined(LZ4_BIG_ENDIAN) */
276 #if defined(_MSC_VER) && !defined(LZ4_FORCE_SW_BITCOUNT)
277         unsigned long r = 0;
278         _BitScanForward64(&r, val);
279         return (int)(r >> 3);
280 #elif defined(__GNUC__) && ((__GNUC__ * 100 + __GNUC_MINOR__) >= 304) && \
281         !defined(LZ4_FORCE_SW_BITCOUNT)
282         return (__builtin_ctzll(val) >> 3);
283 #else
284         static const int DeBruijnBytePos[64] = {
285             0, 0, 0, 0, 0, 1, 1, 2,
286             0, 3, 1, 3, 1, 4, 2, 7,
287             0, 2, 3, 6, 1, 5, 3, 5,
288             1, 3, 4, 4, 2, 5, 6, 7,
289             7, 0, 1, 2, 3, 3, 4, 6,
290             2, 6, 5, 5, 3, 4, 5, 6,
291             7, 1, 2, 4, 6, 4, 4, 5,
292             7, 2, 6, 5, 7, 6, 7, 7
293         };
294         return DeBruijnBytePos[((U64)((val & -val) * 0x0218A392CDABBD3F)) >>
295             58];
296 #endif
297 #endif /* !defined(LZ4_BIG_ENDIAN) */
298 }

300 #else /* !LZ4_ARCH64 */

302 static inline int
303 LZ4_NbCommonBytes(register U32 val)
304 {
305 #if defined(LZ4_BIG_ENDIAN)
306 #if defined(_MSC_VER) && !defined(LZ4_FORCE_SW_BITCOUNT)
307         unsigned long r = 0;
308         _BitScanReverse(&r, val);
309         return (int)(r >> 3);
310 #elif defined(__GNUC__) && ((__GNUC__ * 100 + __GNUC_MINOR__) >= 304) && \
311         !defined(LZ4_FORCE_SW_BITCOUNT)
312         return (__builtin_clz(val) >> 3);
313 #else
314         int r;
315         if (!(val >> 16)) {
316                 r = 2;
317                 val >>= 8;
318         } else {
319                 r = 0;
320                 val >>= 24;
321         }
322         r += (!val);
323         return (r);
324 #endif
325 #else /* !defined(LZ4_BIG_ENDIAN) */
```



```
326 #if defined(_MSC_VER) && !defined(LZ4_FORCE_SW_BITCOUNT)
327         unsigned long r = 0;
328         _BitScanForward(&r, val);
329         return (int)(r >> 3);
330 #elif defined(__GNUC__) && ((__GNUC__ * 100 + __GNUC_MINOR__) >= 304) && \
331         !defined(LZ4_FORCE_SW_BITCOUNT)
332         return (__builtin_ctz(val) >> 3);
333 #else
334         static const int DeBruijnBytePos[32] = {
335             0, 0, 3, 0, 3, 1, 3, 0,
336             3, 2, 2, 1, 3, 2, 0, 1,
337             3, 3, 1, 2, 2, 2, 2, 0,
338             3, 1, 2, 0, 1, 0, 1, 1
339         };
340         return (DeBruijnBytePos[((U32)((val & -(S32)val) * 0x077CB531U)) >>
341             27]);
342 #endif
343 #endif /* !defined(LZ4_BIG_ENDIAN) */
344 }

346 #endif /* !LZ4_ARCH64 */

348 static inline void
349 LZ4HC_Init(LZ4HC_Data_Structure *hc4, const BYTE *base)
350 {
351         MEM_INIT((void *)hc4->hashTable, 0, sizeof (hc4->hashTable));
352         MEM_INIT(hc4->chainTable, 0xFF, sizeof (hc4->chainTable));
353         hc4->nextToUpdate = base + LZ4_ARCH64;
354         hc4->base = base;
355 }

357 static inline void *
358 LZ4HC_Create(const BYTE *base)
359 {
360         LZ4HC_Data_Structure *hc4 = ALLOCATOR(sizeof (LZ4HC_Data_Structure));

362         LZ4HC_Init(hc4, base);
363         return (hc4);
364 }

366 static inline void
367 LZ4HC_Free(LZ4HC_Data_Structure **LZ4HC_Data)
368 {
369         FREEMEM(*LZ4HC_Data, sizeof (LZ4HC_Data_Structure));
370         *LZ4HC_Data = NULL;
371 }


374 static inline void
375 LZ4HC_Insert(LZ4HC_Data_Structure *hc4, const BYTE *ip)
376 {
377         U16 *chainTable = hc4->chainTable;
378         HTYPE *HashTable = hc4->hashTable;
379         INITBASE(base, hc4->base);

381         while (hc4->nextToUpdate < ip) {
382                 ADD_HASH(hc4->nextToUpdate);
383                 hc4->nextToUpdate++;
384         }
385 }


388 static inline int
389 LZ4HC_InsertAndFindBestMatch(LZ4HC_Data_Structure *hc4, const BYTE *ip,
390     const BYTE *const matchlimit, const BYTE **matchpos)
391 {
```

```
392         U16 *const chainTable = hc4->chainTable;
393         HTYPE *const HashTable = hc4->hashTable;
394         const BYTE *ref;
395         INITBASE(base, hc4->base);
396         int nbAttempts = MAX_NB_ATTEMPTS;
397         int ml = 0;

399         /* HC4 match finder */
400         LZ4HC_Insert(hc4, ip);
401         ref = HASH_POINTER(ip);
402         while ((ref >= (ip - MAX_DISTANCE)) && (nbAttempts)) {
403                 nbAttempts--;
404                 if (*(ref + ml) == *(ip + ml))
405                 if (A32(ref) == A32(ip)) {
406                         const BYTE *reft = ref + MINMATCH;
407                         const BYTE *ipt = ip + MINMATCH;

409                         while (ipt < matchlimit - (STEPSIZE - 1)) {
410                                 UARCH diff = AARCH(reft) ^ AARCH(ipt);
411                                 if (!diff) {
412                                         ipt += STEPSIZE;
413                                         reft += STEPSIZE;
414                                         continue;
415                                 }
416                                 ipt += LZ4_NbCommonBytes(diff);
417                                 goto _endCount;
418                         }
419 #if LZ4_ARCH64
420                         if ((ipt < (matchlimit - 3)) &&
421                             (A32(reft) == A32(ipt))) {
422                                 ipt += 4;
423                                 reft += 4;
424                         }
425 #endif /* LZ4_ARCH64 */
426                         if ((ipt < (matchlimit - 1)) &&
427                             (A16(reft) == A16(ipt))) {
428                                 ipt += 2;
429                                 reft += 2;
430                         }
431                         if ((ipt < matchlimit) && (*reft == *ipt))
432                                 ipt++;
433                 _endCount:
434                         if (ipt - ip > ml) {
435                                 ml = (int)(ipt - ip);
436                                 *matchpos = ref;
437                         }
438                 }
439                 ref = GETNEXT(ref);
440         }

442         return (ml);
443 }


446 static inline int
447 LZ4HC_InsertAndGetWiderMatch(LZ4HC_Data_Structure *hc4, const BYTE *ip,
448     const BYTE *startLimit, const BYTE *matchlimit, int longest,
449     const BYTE **matchpos, const BYTE **startpos)
450 {
451         U16 *const chainTable = hc4->chainTable;
452         HTYPE *const HashTable = hc4->hashTable;
453         INITBASE(base, hc4->base);
454         const BYTE *ref;
455         int nbAttempts = MAX_NB_ATTEMPTS;
456         int delta = (int)(ip - startLimit);
```



```
458         /* First Match */
459         LZ4HC_Insert(hc4, ip);
460         ref = HASH_POINTER(ip);

462         while ((ref >= ip-MAX_DISTANCE) && (ref >= hc4->base) && (nbAttempts)) {
463                 nbAttempts--;
464                 if (*(startLimit + longest) == *(ref - delta + longest))
465                         if (A32(ref) == A32(ip)) {
466                                 const BYTE *reft = ref + MINMATCH;
467                                 const BYTE *ipt = ip + MINMATCH;
468                                 const BYTE *startt = ip;

470                                 while (ipt < matchlimit - (STEPSIZE - 1)) {
471                                         UARCH diff = AARCH(reft) ^ AARCH(ipt);
472                                         if (!diff) {
473                                                 ipt += STEPSIZE;
474                                                 reft += STEPSIZE;
475                                                 continue;
476                                         }
477                                         ipt += LZ4_NbCommonBytes(diff);
478                                         goto _endCount;
479                                 }
480 #if LZ4_ARCH64
481                                 if ((ipt < (matchlimit - 3)) &&
482                                     (A32(reft) == A32(ipt))) {
483                                         ipt += 4;
484                                         reft += 4;
485                                 }
486 #endif /* LZ4_ARCH64 */
487                                 if ((ipt < (matchlimit - 1)) &&
488                                     (A16(reft) == A16(ipt))) {
489                                         ipt += 2;
490                                         reft += 2;
491                                 }
492                                 if ((ipt < matchlimit) && (*reft == *ipt))
493                                         ipt++;

495                         _endCount:
496                                 reft = ref;
497                                 while ((startt > startLimit) &&
498                                     (reft > hc4->base) &&
499                                     (startt[-1] == reft[-1])) {
500                                         startt--;
501                                         reft--;
502                                 }

504                                 if ((ipt-startt) > longest) {
505                                         longest = (int)(ipt-startt);
506                                         *matchpos = reft;
507                                         *startpos = startt;
508                                 }
509                         }
510                 ref = GETNEXT(ref);
511         }

513         return (longest);
514 }

516 static inline int
517 LZ4_encodeSequence(const BYTE **ip, BYTE **op, const BYTE *oend,
518     const BYTE **anchor, int ml, const BYTE *ref)
519 {
520         int length, len;
521         BYTE *token;

523         /* Encode Literal length */
```

```
524         length = (int)(*ip - *anchor);
525         token = (*op)++;
527         /* Check output limit */
528         if unlikely((*op) + length + (2 + 1 + LASTLITERALS) + (length >> 8) >
529             oend)
530                 return (0);
532         if (length >= (int) RUN_MASK) {
533                 *token = (RUN_MASK << ML_BITS);
534                 len = length - RUN_MASK;
535                 for (; len > 254; len -= 255)
536                         *(*op)++ = 255;
537                 *(*op)++ = (BYTE)len;
538         } else
539                 *token = (length << ML_BITS);
541         /* Copy Literals */
542         LZ4_BLINDCOPY(*anchor, *op, length);
544         /* Encode Offset */
545         LZ4_WRITE_LITTLEENDIAN_16(*op, (U16)(*ip - ref));
547         /* Encode MatchLength */
548         len = (int)(ml - MINMATCH);
549         /* Check output limit */
550         if unlikely((*op) + (1 + LASTLITERALS) + (len >> 8) > oend)
551                 return (0);
552         if (len >= (int)ML_MASK) {
553                 *token += ML_MASK;
554                 len -= ML_MASK;
555                 for (; len > 509; len -= 510) {
556                         *(*op)++ = 255;
557                         *(*op)++ = 255;
558                 }
559                 if (len > 254) {
560                         len -= 255;
561                         *(*op)++ = 255;
562                 }
563                 *(*op)++ = (BYTE)len;
564         } else
565                 *token += len;
567         /* Prepare next loop */
568         *ip += ml;
569         *anchor = *ip;
571         return (1);
572 }
574 /* Compression CODE */
576 static int
577 LZ4_compressHCCtx(LZ4HC_Data_Structure *ctx, const char *source, char *dest,
578     int isize, int osize)
579 {
580         const BYTE *ip = (const BYTE*) source;
581         const BYTE *anchor = ip;
582         const BYTE *const iend = ip + isize;
583         const BYTE *const oend = (BYTE *) dest + osize;
584         const BYTE *const mflimit = iend - MFLIMIT;
585         const BYTE *const matchlimit = (iend - LASTLITERALS);
587         BYTE *op = (BYTE*) dest;
589         int ml, ml2, ml3, ml0;
```



```
590         const BYTE *ref = NULL;
591         const BYTE *start2 = NULL;
592         const BYTE *ref2 = NULL;
593         const BYTE *start3 = NULL;
594         const BYTE *ref3 = NULL;
595         const BYTE *start0;
596         const BYTE *ref0;
598         ip++;
600         /* Main Loop */
601         while (ip < mflimit) {
602                 ml = LZ4HC_InsertAndFindBestMatch(ctx, ip, matchlimit, (&ref));
603                 if (!ml) {
604                         ip++;
605                         continue;
606                 }
608                 /* saved, in case we would skip too much */
609                 start0 = ip;
610                 ref0 = ref;
611                 ml0 = ml;
613         _Search2:
614                 if (ip+ml < mflimit)
615                         ml2 = LZ4HC_InsertAndGetWiderMatch(ctx, ip + ml - 2,
616                             ip + 1, matchlimit, ml, &ref2, &start2);
617                 else
618                         ml2 = ml;
620                 /* No better match */
621                 if (ml2 == ml) {
622                         if unlikely(!LZ4_encodeSequence(&ip, &op, oend,
623                             &anchor, ml, ref))
624                                 return (0);
625                         continue;
626                 }
628                 if (start0 < ip) {
629                         /* empirical */
630                         if (start2 < ip + ml0) {
631                                 ip = start0;
632                                 ref = ref0;
633                                 ml = ml0;
634                         }
635                 }
637                 /* Here, start0 == ip */
638                 if ((start2 - ip) < 3) {
639                         /* First Match too small : removed */
640                         ml = ml2;
641                         ip = start2;
642                         ref = ref2;
643                         goto _Search2;
644                 }
646         _Search3:
647                 /*
648                  * Currently we have:
649                  * ml2 > ml1, and
650                  * ip1+3 <= ip2 (usually < ip1+ml1)
651                  */
652                 if ((start2 - ip) < OPTIMAL_ML) {
653                         int correction;
654                         int new_ml = ml;
655                         if (new_ml > OPTIMAL_ML)
```

```
656                             new_ml = OPTIMAL_ML;
657                     if (ip+new_ml > start2 + ml2 - MINMATCH)
658                             new_ml = (int)(start2 - ip) + ml2 - MINMATCH;
659                     correction = new_ml - (int)(start2 - ip);
660                     if (correction > 0) {
661                             start2 += correction;
662                             ref2 += correction;
663                             ml2 -= correction;
664                     }
665             }
666             /*
667              * Now, we have start2 = ip+new_ml, with
668              * new_ml=min(ml, OPTIMAL_ML=18)
669              */

671             if (start2 + ml2 < mflimit)
672                     ml3 = LZ4HC_InsertAndGetWiderMatch(ctx, start2 + ml2 -
673                         3, start2, matchlimit, ml2, &ref3, &start3);
674             else
675                     ml3 = ml2;

677             /* No better match: 2 sequences to encode */
678             if (ml3 == ml2) {
679                     /* ip & ref are known; Now for ml */
680                     if (start2 < ip+ml) {
681                             if ((start2 - ip) < OPTIMAL_ML) {
682                                     int correction;
683                                     if (ml > OPTIMAL_ML) ml = OPTIMAL_ML;
684                                     if (ip+ml > start2 + ml2 - MINMATCH)
685                                             ml = (int)(start2 - ip) +
686                                                 ml2 - MINMATCH;
687                                     correction = ml - (int)(start2 - ip);
688                                     if (correction > 0) {
689                                             start2 += correction;
690                                             ref2 += correction;
691                                             ml2 -= correction;
692                                     }
693                             } else {
694                                     ml = (int)(start2 - ip);
695                             }
696                     }
697                     /* Now, encode 2 sequences */
698                     if unlikely(!LZ4_encodeSequence(&ip, &op, oend,
699                         &anchor, ml, ref))
700                             return (0);
701                     ip = start2;
702                     if unlikely(!LZ4_encodeSequence(&ip, &op, oend,
703                         &anchor, ml2, ref2))
704                             return (0);
705                     continue;
706             }

708             /* Not enough space for match 2: remove it */
709             if (start3 < ip + ml + 3) {
710                     /*
711                      * can write Seq1 immediately ==> Seq2 is removed,
712                      * so Seq3 becomes Seq1
713                      */
714                     if (start3 >= (ip+ml)) {
715                             if (start2 < ip+ml) {
716                                     int correction = (int)(ip+ml - start2);
717                                     start2 += correction;
718                                     ref2 += correction;
719                                     ml2 -= correction;
720                                     if (ml2 < MINMATCH) {
721                                             start2 = start3;
```



```
722                                             ref2 = ref3;
723                                             ml2 = ml3;
724                                     }
725                             }

727                             if unlikely(!LZ4_encodeSequence(&ip, &op, oend,
728                                 &anchor, ml, ref))
729                                     return (0);
730                             ip  = start3;
731                             ref = ref3;
732                             ml  = ml3;

734                             start0 = start2;
735                             ref0 = ref2;
736                             ml0 = ml2;
737                             goto _Search2;
738                     }

740                     start2 = start3;
741                     ref2 = ref3;
742                     ml2 = ml3;
743                     goto _Search3;
744             }

746             /*
747              * OK, now we have 3 ascending matches; let's write at least
748              * the first one
749              * ip & ref are known; Now for ml
750              */
751             if (start2 < ip+ml) {
752                     if ((start2 - ip) < (int)ML_MASK) {
753                             int correction;
754                             if (ml > OPTIMAL_ML)
755                                     ml = OPTIMAL_ML;
756                             if (ip + ml > start2 + ml2 - MINMATCH)
757                                     ml = (int)(start2 - ip) + ml2 -
758                                         MINMATCH;
759                             correction = ml - (int)(start2 - ip);
760                             if (correction > 0) {
761                                     start2 += correction;
762                                     ref2 += correction;
763                                     ml2 -= correction;
764                             }
765                     } else {
766                             ml = (int)(start2 - ip);
767                     }
768             }
769             if unlikely(!LZ4_encodeSequence(&ip, &op, oend, &anchor, ml,
770                 ref))
771                     return (0);

773             ip = start2;
774             ref = ref2;
775             ml = ml2;

777             start2 = start3;
778             ref2 = ref3;
779             ml2 = ml3;

781             goto _Search3;

783     }

785     /* Encode Last Literals */
786     {
787             int lastRun = (int)(iend - anchor);
```

```
788                 if (op + lastRun + 1 + ((lastRun + 255 - RUN_MASK) / 255) >
789                     oend)
790                         return (0);
791                 if (lastRun >= (int) RUN_MASK) {
792                         *op++ = (RUN_MASK << ML_BITS);
793                         lastRun -= RUN_MASK;
794                         for (; lastRun > 254; lastRun -= 255) {
795                                 *op++ = 255;
796                         }
797                         *op++ = (BYTE) lastRun;
798                 } else
799                         *op++ = (lastRun << ML_BITS);
800                 (void) memcpy(op, anchor, iend - anchor);
801                 op += iend - anchor;
802         }

804         /* End */
805         return (int) (((char *) op) - dest);
806 }

808 int
809 zfs_LZ4_compressHC(const char *source, char *dest, int isize, int osize)
810 {
811         LZ4HC_Data_Structure *ctx = LZ4HC_Create((const BYTE*) source);
812         int result = LZ4_compressHCCtx(ctx, source, dest, isize, osize);
813         LZ4HC_Free(&ctx);

815         return (result);
816 }
```

****
2077 Tue Jan 15 15:38:24 2013
new/usr/src/uts/common/fs/zfs/sys/lz4hc.h
%B
****

```
/*
 * LZ4 HC - High Compression Mode of LZ4
 * Header File
 * Copyright (C) 2011-2012, Yann Collet.
 * BSD 2-Clause License (http://www.opensource.org/licenses/bsd-license.php)
 *
 * Redistribution and use in source and binary forms, with or without
 * modification, are permitted provided that the following conditions are
 * met:
 *
 *     * Redistributions of source code must retain the above copyright
 * notice, this list of conditions and the following disclaimer.
 *     * Redistributions in binary form must reproduce the above
 * copyright notice, this list of conditions and the following disclaimer
 * in the documentation and/or other materials provided with the
 * distribution.
 *
 * THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS
 * "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT
 * LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR
 * A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT
 * OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL,
 * SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT
 * LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE,
 * DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY
 * THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT
 * (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE
 * OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.
 *
 * You can contact the author at :
 * - LZ4 homepage : http://fastcompression.blogspot.com/p/lz4.html
 * - LZ4 source repository : http://code.google.com/p/lz4/
 */
#if defined(__cplusplus)
extern "C" {
#endif

/*
 * LZ4_compressHC :
 *   return : the number of bytes in compressed buffer dest, zero if
 *            compression failed (osize too small)
 */
int zfs_LZ4_compressHC(const char *source, char *dest, int isize, int osize);

/*
 * Note:
 * Decompression functions are provided within regular LZ4 source code
 * (see "lz4.c") (BSD license)
 */

#if defined(__cplusplus)
}
#endif
```

```
**********************************************************
   18027 Tue Jan 15 15:38:24 2013
new/usr/src/uts/common/fs/zfs/sys/zio.h
%B
**********************************************************
```

```
   1 /*
   2  * CDDL HEADER START
   3  *
   4  * The contents of this file are subject to the terms of the
   5  * Common Development and Distribution License (the "License").
   6  * You may not use this file except in compliance with the License.
   7  *
   8  * You can obtain a copy of the license at usr/src/OPENSOLARIS.LICENSE
   9  * or http://www.opensolaris.org/os/licensing.
  10  * See the License for the specific language governing permissions
  11  * and limitations under the License.
  12  *
  13  * When distributing Covered Code, include this CDDL HEADER in each
  14  * file and include the License file at usr/src/OPENSOLARIS.LICENSE.
  15  * If applicable, add the following below this CDDL HEADER, with the
  16  * fields enclosed by brackets "[]" replaced with your own identifying
  17  * information: Portions Copyright [yyyy] [name of copyright owner]
  18  *
  19  * CDDL HEADER END
  20  */

  22 /*
  23  * Copyright (c) 2005, 2010, Oracle and/or its affiliates. All rights reserved.
  24  * Copyright 2011 Nexenta Systems, Inc.  All rights reserved.
  25  * Copyright (c) 2012 by Delphix. All rights reserved.
  26  * Copyright (c) 2013 by Saso Kiselkov. All rights reserved.
  27  */

  29 #ifndef _ZIO_H
  30 #define _ZIO_H

  32 #include <sys/zfs_context.h>
  33 #include <sys/spa.h>
  34 #include <sys/txg.h>
  35 #include <sys/avl.h>
  36 #include <sys/fs/zfs.h>
  37 #include <sys/zio_impl.h>

  39 #ifdef  __cplusplus
  40 extern "C" {
  41 #endif

  43 /*
  44  * Embedded checksum
  45  */
  46 #define ZEC_MAGIC       0x210da7ab10c7a11ULL

  48 typedef struct zio_eck {
  49         uint64_t        zec_magic;      /* for validation, endianness   */
  50         zio_cksum_t     zec_cksum;      /* 256-bit checksum             */
  51 } zio_eck_t;
______unchanged_portion_omitted_

  85 #define ZIO_CHECKSUM_ON_VALUE   ZIO_CHECKSUM_FLETCHER_4
  86 #define ZIO_CHECKSUM_DEFAULT    ZIO_CHECKSUM_ON

  88 #define ZIO_CHECKSUM_MASK       0xffULL
  89 #define ZIO_CHECKSUM_VERIFY     (1 << 8)

  91 #define ZIO_DEDUPCHECKSUM       ZIO_CHECKSUM_SHA256
  92 #define ZIO_DEDUPDITTO_MIN      100
```



```
  94 enum zio_compress {
  95         ZIO_COMPRESS_INHERIT = 0,
  96         ZIO_COMPRESS_ON,
  97         ZIO_COMPRESS_OFF,
  98         ZIO_COMPRESS_LZJB,
  99         ZIO_COMPRESS_EMPTY,
 100         ZIO_COMPRESS_GZIP_1,
 101         ZIO_COMPRESS_GZIP_2,
 102         ZIO_COMPRESS_GZIP_3,
 103         ZIO_COMPRESS_GZIP_4,
 104         ZIO_COMPRESS_GZIP_5,
 105         ZIO_COMPRESS_GZIP_6,
 106         ZIO_COMPRESS_GZIP_7,
 107         ZIO_COMPRESS_GZIP_8,
 108         ZIO_COMPRESS_GZIP_9,
 109         ZIO_COMPRESS_ZLE,
 110         ZIO_COMPRESS_LZ4,
 111         ZIO_COMPRESS_LZ4HC,
 112         ZIO_COMPRESS_FUNCTIONS
 113 };

 115 /* N.B. when altering this value, also change BOOTFS_COMPRESS_VALID below */
 116 #define ZIO_COMPRESS_ON_VALUE   ZIO_COMPRESS_LZJB
 117 #define ZIO_COMPRESS_DEFAULT    ZIO_COMPRESS_OFF

 119 #define BOOTFS_COMPRESS_VALID(compress)                 \
 120         ((compress) == ZIO_COMPRESS_LZJB ||             \
 121         (compress) == ZIO_COMPRESS_LZ4HC ||             \
 122         (compress) == ZIO_COMPRESS_LZ4 ||               \
 123         ((compress) == ZIO_COMPRESS_ON &&               \
 124         ZIO_COMPRESS_ON_VALUE == ZIO_COMPRESS_LZJB) ||  \
 125         (compress) == ZIO_COMPRESS_OFF)

 127 #define ZIO_FAILURE_MODE_WAIT           0
 128 #define ZIO_FAILURE_MODE_CONTINUE       1
 129 #define ZIO_FAILURE_MODE_PANIC          2

 131 #define ZIO_PRIORITY_NOW                (zio_priority_table[0])
 132 #define ZIO_PRIORITY_SYNC_READ          (zio_priority_table[1])
 133 #define ZIO_PRIORITY_SYNC_WRITE         (zio_priority_table[2])
 134 #define ZIO_PRIORITY_LOG_WRITE          (zio_priority_table[3])
 135 #define ZIO_PRIORITY_CACHE_FILL         (zio_priority_table[4])
 136 #define ZIO_PRIORITY_AGG                (zio_priority_table[5])
 137 #define ZIO_PRIORITY_FREE               (zio_priority_table[6])
 138 #define ZIO_PRIORITY_ASYNC_WRITE        (zio_priority_table[7])
 139 #define ZIO_PRIORITY_ASYNC_READ         (zio_priority_table[8])
 140 #define ZIO_PRIORITY_RESILVER           (zio_priority_table[9])
 141 #define ZIO_PRIORITY_SCRUB              (zio_priority_table[10])
 142 #define ZIO_PRIORITY_DDT_PREFETCH       (zio_priority_table[11])
 143 #define ZIO_PRIORITY_TABLE_SIZE         12

 145 #define ZIO_PIPELINE_CONTINUE           0x100
 146 #define ZIO_PIPELINE_STOP               0x101

 148 enum zio_flag {
 149         /*
 150          * Flags inherited by gang, ddt, and vdev children,
 151          * and that must be equal for two zios to aggregate
 152          */
 153         ZIO_FLAG_DONT_AGGREGATE = 1 << 0,
 154         ZIO_FLAG_IO_REPAIR      = 1 << 1,
 155         ZIO_FLAG_SELF_HEAL      = 1 << 2,
 156         ZIO_FLAG_RESILVER       = 1 << 3,
 157         ZIO_FLAG_SCRUB          = 1 << 4,
 158         ZIO_FLAG_SCAN_THREAD    = 1 << 5,
```

```
160 #define ZIO_FLAG_AGG_INHERIT    (ZIO_FLAG_CANFAIL - 1)

162         /*
163          * Flags inherited by ddt, gang, and vdev children.
164          */
165         ZIO_FLAG_CANFAIL        = 1 << 6,       /* must be first for INHERIT */
166         ZIO_FLAG_SPECULATIVE    = 1 << 7,
167         ZIO_FLAG_CONFIG_WRITER  = 1 << 8,
168         ZIO_FLAG_DONT_RETRY     = 1 << 9,
169         ZIO_FLAG_DONT_CACHE     = 1 << 10,
170         ZIO_FLAG_NODATA         = 1 << 11,
171         ZIO_FLAG_INDUCE_DAMAGE  = 1 << 12,

173 #define ZIO_FLAG_DDT_INHERIT    (ZIO_FLAG_IO_RETRY - 1)
174 #define ZIO_FLAG_GANG_INHERIT   (ZIO_FLAG_IO_RETRY - 1)

176         /*
177          * Flags inherited by vdev children.
178          */
179         ZIO_FLAG_IO_RETRY       = 1 << 13,      /* must be first for INHERIT */
180         ZIO_FLAG_PROBE          = 1 << 14,
181         ZIO_FLAG_TRYHARD        = 1 << 15,
182         ZIO_FLAG_OPTIONAL       = 1 << 16,

184 #define ZIO_FLAG_VDEV_INHERIT   (ZIO_FLAG_DONT_QUEUE - 1)

186         /*
187          * Flags not inherited by any children.
188          */
189         ZIO_FLAG_DONT_QUEUE     = 1 << 17,      /* must be first for INHERIT */
190         ZIO_FLAG_DONT_PROPAGATE = 1 << 18,
191         ZIO_FLAG_IO_BYPASS      = 1 << 19,
192         ZIO_FLAG_IO_REWRITE     = 1 << 20,
193         ZIO_FLAG_RAW            = 1 << 21,
194         ZIO_FLAG_GANG_CHILD     = 1 << 22,
195         ZIO_FLAG_DDT_CHILD      = 1 << 23,
196         ZIO_FLAG_GODFATHER      = 1 << 24,
197         ZIO_FLAG_NOPWRITE       = 1 << 25,
198         ZIO_FLAG_REEXECUTED     = 1 << 26,
199 };
```

*unchanged_portion_omitted_*

```
**********************************************************
    2863 Tue Jan 15 15:38:24 2013
new/usr/src/uts/common/fs/zfs/sys/zio_compress.h
%B
**********************************************************
```

```
/*
 * CDDL HEADER START
 *
 * The contents of this file are subject to the terms of the
 * Common Development and Distribution License (the "License").
 * You may not use this file except in compliance with the License.
 *
 * You can obtain a copy of the license at usr/src/OPENSOLARIS.LICENSE
 * or http://www.opensolaris.org/os/licensing.
 * See the License for the specific language governing permissions
 * and limitations under the License.
 *
 * When distributing Covered Code, include this CDDL HEADER in each
 * file and include the License file at usr/src/OPENSOLARIS.LICENSE.
 * If applicable, add the following below this CDDL HEADER, with the
 * fields enclosed by brackets "[]" replaced with your own identifying
 * information: Portions Copyright [yyyy] [name of copyright owner]
 *
 * CDDL HEADER END
 */

/*
 * Copyright 2009 Sun Microsystems, Inc.  All rights reserved.
 * Use is subject to license terms.
 */
/*
 * Copyright (c) 2013 by Saso Kiselkov. All rights reserved.
 */

#ifndef _SYS_ZIO_COMPRESS_H
#define _SYS_ZIO_COMPRESS_H

#include <sys/zio.h>

#ifdef  __cplusplus
extern "C" {
#endif

/*
 * Common signature for all zio compress/decompress functions.
 */
typedef size_t zio_compress_func_t(void *src, void *dst,
    size_t s_len, size_t d_len, int);
typedef int zio_decompress_func_t(void *src, void *dst,
    size_t s_len, size_t d_len, int);

/*
 * Information about each compression function.
 */
typedef struct zio_compress_info {
        zio_compress_func_t     *ci_compress;   /* compression function */
        zio_decompress_func_t   *ci_decompress; /* decompression function */
        int                     ci_level;       /* level parameter */
        char                    *ci_name;       /* algorithm name */
} zio_compress_info_t;

extern zio_compress_info_t zio_compress_table[ZIO_COMPRESS_FUNCTIONS];

/*
 * Compression routines.
 */
```



```
extern size_t lzjb_compress(void *src, void *dst, size_t s_len, size_t d_len,
    int level);
extern int lzjb_decompress(void *src, void *dst, size_t s_len, size_t d_len,
    int level);
extern size_t gzip_compress(void *src, void *dst, size_t s_len, size_t d_len,
    int level);
extern int gzip_decompress(void *src, void *dst, size_t s_len, size_t d_len,
    int level);
extern size_t zle_compress(void *src, void *dst, size_t s_len, size_t d_len,
    int level);
extern int zle_decompress(void *src, void *dst, size_t s_len, size_t d_len,
    int level);
extern size_t lz4_compress(void *src, void *dst, size_t s_len, size_t d_len,
    int level);
extern int lz4_decompress(void *src, void *dst, size_t s_len, size_t d_len,
    int level);

/*
 * Compress and decompress data if necessary.
 */
extern size_t zio_compress_data(enum zio_compress c, void *src, void *dst,
    size_t s_len);
extern int zio_decompress_data(enum zio_compress c, void *src, void *dst,
    size_t s_len, size_t d_len);

#ifdef  __cplusplus
}
```

*unchanged_portion_omitted_*

```
**********************************************************
  145352 Tue Jan 15 15:38:24 2013
new/usr/src/uts/common/fs/zfs/zfs_ioctl.c
%B
**********************************************************
```

```
   1 /*
   2  * CDDL HEADER START
   3  *
   4  * The contents of this file are subject to the terms of the
   5  * Common Development and Distribution License (the "License").
   6  * You may not use this file except in compliance with the License.
   7  *
   8  * You can obtain a copy of the license at usr/src/OPENSOLARIS.LICENSE
   9  * or http://www.opensolaris.org/os/licensing.
  10  * See the License for the specific language governing permissions
  11  * and limitations under the License.
  12  *
  13  * When distributing Covered Code, include this CDDL HEADER in each
  14  * file and include the License file at usr/src/OPENSOLARIS.LICENSE.
  15  * If applicable, add the following below this CDDL HEADER, with the
  16  * fields enclosed by brackets "[]" replaced with your own identifying
  17  * information: Portions Copyright [yyyy] [name of copyright owner]
  18  *
  19  * CDDL HEADER END
  20  */

  22 /*
  23  * Copyright (c) 2005, 2010, Oracle and/or its affiliates. All rights reserved.
  24  * Portions Copyright 2011 Martin Matuska
  25  * Copyright 2011 Nexenta Systems, Inc.  All rights reserved.
  26  * Copyright (c) 2012, Joyent, Inc. All rights reserved.
  27  * Copyright (c) 2012 by Delphix. All rights reserved.
  28  * Copyright (c) 2013 by Saso Kiselkov. All rights reserved.
  29  */

  31 /*
  32  * ZFS ioctls.
  33  *
  34  * This file handles the ioctls to /dev/zfs, used for configuring ZFS storage
  35  * pools and filesystems, e.g. with /sbin/zfs and /sbin/zpool.
  36  *
  37  * There are two ways that we handle ioctls: the legacy way where almost
  38  * all of the logic is in the ioctl callback, and the new way where most
  39  * of the marshalling is handled in the common entry point, zfsdev_ioctl().
  40  *
  41  * Non-legacy ioctls should be registered by calling
  42  * zfs_ioctl_register() from zfs_ioctl_init().  The ioctl is invoked
  43  * from userland by lzc_ioctl().
  44  *
  45  * The registration arguments are as follows:
  46  *
  47  * const char *name
  48  *   The name of the ioctl.  This is used for history logging.  If the
  49  *   ioctl returns successfully (the callback returns 0), and allow_log
  50  *   is true, then a history log entry will be recorded with the input &
  51  *   output nvlists.  The log entry can be printed with "zpool history -i".
  52  *
  53  * zfs_ioc_t ioc
  54  *   The ioctl request number, which userland will pass to ioctl(2).
  55  *   The ioctl numbers can change from release to release, because
  56  *   the caller (libzfs) must be matched to the kernel.
  57  *
  58  * zfs_secpolicy_func_t *secpolicy
  59  *   This function will be called before the zfs_ioc_func_t, to
  60  *   determine if this operation is permitted.  It should return EPERM
  61  *   on failure, and 0 on success.  Checks include determining if the
```



```
  62  *   dataset is visible in this zone, and if the user has either all
  63  *   zfs privileges in the zone (SYS_MOUNT), or has been granted permission
  64  *   to do this operation on this dataset with "zfs allow".
  65  *
  66  * zfs_ioc_namecheck_t namecheck
  67  *   This specifies what to expect in the zfs_cmd_t:zc_name -- a pool
  68  *   name, a dataset name, or nothing.  If the name is not well-formed,
  69  *   the ioctl will fail and the callback will not be called.
  70  *   Therefore, the callback can assume that the name is well-formed
  71  *   (e.g. is null-terminated, doesn't have more than one '@' character,
  72  *   doesn't have invalid characters).
  73  *
  74  * zfs_ioc_poolcheck_t pool_check
  75  *   This specifies requirements on the pool state.  If the pool does
  76  *   not meet them (is suspended or is readonly), the ioctl will fail
  77  *   and the callback will not be called.  If any checks are specified
  78  *   (i.e. it is not POOL_CHECK_NONE), namecheck must not be NO_NAME.
  79  *   Multiple checks can be or-ed together (e.g. POOL_CHECK_SUSPENDED |
  80  *   POOL_CHECK_READONLY).
  81  *
  82  * boolean_t smush_outnvlist
  83  *   If smush_outnvlist is true, then the output is presumed to be a
  84  *   list of errors, and it will be "smushed" down to fit into the
  85  *   caller's buffer, by removing some entries and replacing them with a
  86  *   single "N_MORE_ERRORS" entry indicating how many were removed.  See
  87  *   nvlist_smush() for details.  If smush_outnvlist is false, and the
  88  *   outnvlist does not fit into the userland-provided buffer, then the
  89  *   ioctl will fail with ENOMEM.
  90  *
  91  * zfs_ioc_func_t *func
  92  *   The callback function that will perform the operation.
  93  *
  94  *   The callback should return 0 on success, or an error number on
  95  *   failure.  If the function fails, the userland ioctl will return -1,
  96  *   and errno will be set to the callback's return value.  The callback
  97  *   will be called with the following arguments:
  98  *
  99  *   const char *name
 100  *     The name of the pool or dataset to operate on, from
 101  *     zfs_cmd_t:zc_name.  The 'namecheck' argument specifies the
 102  *     expected type (pool, dataset, or none).
 103  *
 104  *   nvlist_t *innvl
 105  *     The input nvlist, deserialized from zfs_cmd_t:zc_nvlist_src.  Or
 106  *     NULL if no input nvlist was provided.  Changes to this nvlist are
 107  *     ignored.  If the input nvlist could not be deserialized, the
 108  *     ioctl will fail and the callback will not be called.
 109  *
 110  *   nvlist_t *outnvl
 111  *     The output nvlist, initially empty.  The callback can fill it in,
 112  *     and it will be returned to userland by serializing it into
 113  *     zfs_cmd_t:zc_nvlist_dst.  If it is non-empty, and serialization
 114  *     fails (e.g. because the caller didn't supply a large enough
 115  *     buffer), then the overall ioctl will fail.  See the
 116  *     'smush_nvlist' argument above for additional behaviors.
 117  *
 118  *     There are two typical uses of the output nvlist:
 119  *       - To return state, e.g. property values.  In this case,
 120  *         smush_outnvlist should be false.  If the buffer was not large
 121  *         enough, the caller will reallocate a larger buffer and try
 122  *         the ioctl again.
 123  *
 124  *       - To return multiple errors from an ioctl which makes on-disk
 125  *         changes.  In this case, smush_outnvlist should be true.
 126  *         Ioctls which make on-disk modifications should generally not
 127  *         use the outnvl if they succeed, because the caller can not
```

```
 128  *         distinguish between the operation failing, and
 129  *         deserialization failing.
 130  */

 132 #include <sys/types.h>
 133 #include <sys/param.h>
 134 #include <sys/errno.h>
 135 #include <sys/uio.h>
 136 #include <sys/buf.h>
 137 #include <sys/modctl.h>
 138 #include <sys/open.h>
 139 #include <sys/file.h>
 140 #include <sys/kmem.h>
 141 #include <sys/conf.h>
 142 #include <sys/cmn_err.h>
 143 #include <sys/stat.h>
 144 #include <sys/zfs_ioctl.h>
 145 #include <sys/zfs_vfsops.h>
 146 #include <sys/zfs_znode.h>
 147 #include <sys/zap.h>
 148 #include <sys/spa.h>
 149 #include <sys/spa_impl.h>
 150 #include <sys/vdev.h>
 151 #include <sys/priv_impl.h>
 152 #include <sys/dmu.h>
 153 #include <sys/dsl_dir.h>
 154 #include <sys/dsl_dataset.h>
 155 #include <sys/dsl_prop.h>
 156 #include <sys/dsl_deleg.h>
 157 #include <sys/dmu_objset.h>
 158 #include <sys/dmu_impl.h>
 159 #include <sys/ddi.h>
 160 #include <sys/sunddi.h>
 161 #include <sys/sunldi.h>
 162 #include <sys/policy.h>
 163 #include <sys/zone.h>
 164 #include <sys/nvpair.h>
 165 #include <sys/pathname.h>
 166 #include <sys/mount.h>
 167 #include <sys/sdt.h>
 168 #include <sys/fs/zfs.h>
 169 #include <sys/zfs_ctldir.h>
 170 #include <sys/zfs_dir.h>
 171 #include <sys/zfs_onexit.h>
 172 #include <sys/zvol.h>
 173 #include <sys/dsl_scan.h>
 174 #include <sharefs/share.h>
 175 #include <sys/dmu_objset.h>
 176 #include <sys/zfeature.h>

 178 #include "zfs_namecheck.h"
 179 #include "zfs_prop.h"
 180 #include "zfs_deleg.h"
 181 #include "zfs_comutil.h"

 183 extern struct modlfs zfs_modlfs;

 185 extern void zfs_init(void);
 186 extern void zfs_fini(void);

 188 ldi_ident_t zfs_li = NULL;
 189 dev_info_t *zfs_dip;

 191 uint_t zfs_fsyncer_key;
 192 extern uint_t rrw_tsd_key;
 193 static uint_t zfs_allow_log_key;
```



```
 195 typedef int zfs_ioc_legacy_func_t(zfs_cmd_t *);
 196 typedef int zfs_ioc_func_t(const char *, nvlist_t *, nvlist_t *);
 197 typedef int zfs_secpolicy_func_t(zfs_cmd_t *, nvlist_t *, cred_t *);

 199 typedef enum {
 200         NO_NAME,
 201         POOL_NAME,
 202         DATASET_NAME
 203 } zfs_ioc_namecheck_t;
_____unchanged_portion_omitted_

 230 static int zfs_ioc_userspace_upgrade(zfs_cmd_t *zc);
 231 static int zfs_check_settable(const char *name, nvpair_t *property,
 232     cred_t *cr);
 233 static int zfs_check_clearable(char *dataset, nvlist_t *props,
 234     nvlist_t **errors);
 235 static int zfs_fill_zplprops_root(uint64_t, nvlist_t *, nvlist_t *,
 236     boolean_t *);
 237 int zfs_set_prop_nvlist(const char *, zprop_source_t, nvlist_t *, nvlist_t *);
 238 static int get_nvlist(uint64_t nvl, uint64_t size, int iflag, nvlist_t **nvp);

 240 static int zfs_prop_activate_feature(dsl_pool_t *dp, zfeature_info_t *feature);
 241 static int zfs_prop_activate_feature_check(void *arg1, void *arg2,
 242     dmu_tx_t *tx);
 243 static void zfs_prop_activate_feature_sync(void *arg1, void *arg2,
 244     dmu_tx_t *tx);

 246 /* _NOTE(PRINTFLIKE(4)) - this is printf-like, but lint is too whiney */
 247 void
 248 __dprintf(const char *file, const char *func, int line, const char *fmt, ...)
 249 {
 250         const char *newfile;
 251         char buf[512];
 252         va_list adx;

 254         /*
 255          * Get rid of annoying "../common/" prefix to filename.
 256          */
 257         newfile = strrchr(file, '/');
 258         if (newfile != NULL) {
 259                 newfile = newfile + 1; /* Get rid of leading / */
 260         } else {
 261                 newfile = file;
 262         }

 264         va_start(adx, fmt);
 265         (void) vsnprintf(buf, sizeof (buf), fmt, adx);
 266         va_end(adx);

 268         /*
 269          * To get this data, use the zfs-dprintf probe as so:
 270          * dtrace -q -n 'zfs-dprintf \
 271          *      /stringof(arg0) == "dbuf.c"/ \
 272          *      {printf("%s: %s", stringof(arg1), stringof(arg3))}'
 273          * arg0 = file name
 274          * arg1 = function name
 275          * arg2 = line number
 276          * arg3 = message
 277          */
 278         DTRACE_PROBE4(zfs__dprintf,
 279             char *, newfile, char *, func, int, line, char *, buf);
 280 }
_____unchanged_portion_omitted_

2322 /*
```

```
 * If the named property is one that has a special function to set its value,
 * return 0 on success and a positive error code on failure; otherwise if it is
 * not one of the special properties handled by this function, return -1.
 *
 * XXX: It would be better for callers of the property interface if we handled
 * these special cases in dsl_prop.c (in the dsl layer).
 */
static int
zfs_prop_set_special(const char *dsname, zprop_source_t source,
    nvpair_t *pair)
{
        const char *propname = nvpair_name(pair);
        zfs_prop_t prop = zfs_name_to_prop(propname);
        uint64_t intval;
        int err;

        if (prop == ZPROP_INVAL) {
                if (zfs_prop_userquota(propname))
                        return (zfs_prop_set_userquota(dsname, pair));
                return (-1);
        }

        if (nvpair_type(pair) == DATA_TYPE_NVLIST) {
                nvlist_t *attrs;
                VERIFY(nvpair_value_nvlist(pair, &attrs) == 0);
                VERIFY(nvlist_lookup_nvpair(attrs, ZPROP_VALUE,
                    &pair) == 0);
        }

        if (zfs_prop_get_type(prop) == PROP_TYPE_STRING)
                return (-1);

        VERIFY(0 == nvpair_value_uint64(pair, &intval));

        switch (prop) {
        case ZFS_PROP_QUOTA:
                err = dsl_dir_set_quota(dsname, source, intval);
                break;
        case ZFS_PROP_REFQUOTA:
                err = dsl_dataset_set_quota(dsname, source, intval);
                break;
        case ZFS_PROP_RESERVATION:
                err = dsl_dir_set_reservation(dsname, source, intval);
                break;
        case ZFS_PROP_REFRESERVATION:
                err = dsl_dataset_set_reservation(dsname, source, intval);
                break;
        case ZFS_PROP_VOLSIZE:
                err = zvol_set_volsize(dsname, ddi_driver_major(zfs_dip),
                    intval);
                break;
        case ZFS_PROP_VERSION:
        {
                zfsvfs_t *zfsvfs;

                if ((err = zfsvfs_hold(dsname, FTAG, &zfsvfs, B_TRUE)) != 0)
                        break;

                err = zfs_set_version(zfsvfs, intval);
                zfsvfs_rele(zfsvfs, FTAG);

                if (err == 0 && intval >= ZPL_VERSION_USERSPACE) {
                        zfs_cmd_t *zc;

                        zc = kmem_zalloc(sizeof (zfs_cmd_t), KM_SLEEP);
                        (void) strcpy(zc->zc_name, dsname);
```



```
                        (void) zfs_ioc_userspace_upgrade(zc);
                        kmem_free(zc, sizeof (zfs_cmd_t));
                }
                break;
        }
        case ZFS_PROP_COMPRESSION:
        {
                if (intval == ZIO_COMPRESS_LZ4 ||
                    intval == ZIO_COMPRESS_LZ4HC) {
                        zfeature_info_t *feature =
                            &spa_feature_table[SPA_FEATURE_LZ4_COMPRESS];
                        spa_t *spa;
                        dsl_pool_t *dp;

                        if ((err = spa_open(dsname, &spa, FTAG)) != 0)
                                return (err);

                        dp = spa->spa_dsl_pool;

                        /*
                         * Setting the LZ4 compression algorithm activates
                         * the feature.
                         */
                        if (!spa_feature_is_active(spa, feature)) {
                                if ((err = zfs_prop_activate_feature(dp,
                                    feature)) != 0) {
                                        spa_close(spa, FTAG);
                                        return (err);
                                }
                        }

                        spa_close(spa, FTAG);
                }
                /*
                 * We still want the default set action to be performed in the
                 * caller, we only performed zfeature settings here.
                 */
                err = -1;
                break;
        }

        default:
                err = -1;
        }

        return (err);
}
_____unchanged_portion_omitted_

static int
zfs_check_settable(const char *dsname, nvpair_t *pair, cred_t *cr)
{
        const char *propname = nvpair_name(pair);
        boolean_t issnap = (strchr(dsname, '@') != NULL);
        zfs_prop_t prop = zfs_name_to_prop(propname);
        uint64_t intval;
        int err;

        if (prop == ZPROP_INVAL) {
                if (zfs_prop_user(propname)) {
                        if (err = zfs_secpolicy_write_perms(dsname,
                            ZFS_DELEG_PERM_USERPROP, cr))
                                return (err);
                        return (0);
                }
```

```
3609                 if (!issnap && zfs_prop_userquota(propname)) {
3610                         const char *perm = NULL;
3611                         const char *uq_prefix =
3612                             zfs_userquota_prop_prefixes[ZFS_PROP_USERQUOTA];
3613                         const char *gq_prefix =
3614                             zfs_userquota_prop_prefixes[ZFS_PROP_GROUPQUOTA];
3616                         if (strncmp(propname, uq_prefix,
3617                             strlen(uq_prefix)) == 0) {
3618                                 perm = ZFS_DELEG_PERM_USERQUOTA;
3619                         } else if (strncmp(propname, gq_prefix,
3620                             strlen(gq_prefix)) == 0) {
3621                                 perm = ZFS_DELEG_PERM_GROUPQUOTA;
3622                         } else {
3623                                 /* USERUSED and GROUPUSED are read-only */
3624                                 return (EINVAL);
3625                         }
3627                         if (err = zfs_secpolicy_write_perms(dsname, perm, cr))
3628                                 return (err);
3629                         return (0);
3630                 }
3632                 return (EINVAL);
3633         }
3635         if (issnap)
3636                 return (EINVAL);
3638         if (nvpair_type(pair) == DATA_TYPE_NVLIST) {
3639                 /*
3640                  * dsl_prop_get_all_impl() returns properties in this
3641                  * format.
3642                  */
3643                 nvlist_t *attrs;
3644                 VERIFY(nvpair_value_nvlist(pair, &attrs) == 0);
3645                 VERIFY(nvlist_lookup_nvpair(attrs, ZPROP_VALUE,
3646                     &pair) == 0);
3647         }
3649         /*
3650          * Check that this value is valid for this pool version
3651          */
3652         switch (prop) {
3653         case ZFS_PROP_COMPRESSION:
3654                 /*
3655                  * If the user specified gzip compression, make sure
3656                  * the SPA supports it. We ignore any errors here since
3657                  * we'll catch them later.
3658                  */
3659                 if (nvpair_type(pair) == DATA_TYPE_UINT64 &&
3660                     nvpair_value_uint64(pair, &intval) == 0) {
3661                         if (intval >= ZIO_COMPRESS_GZIP_1 &&
3662                             intval <= ZIO_COMPRESS_GZIP_9 &&
3663                             zfs_earlier_version(dsname,
3664                             SPA_VERSION_GZIP_COMPRESSION)) {
3665                                 return (ENOTSUP);
3666                         }
3668                         if (intval == ZIO_COMPRESS_ZLE &&
3669                             zfs_earlier_version(dsname,
3670                             SPA_VERSION_ZLE_COMPRESSION))
3671                                 return (ENOTSUP);
3673                         if (intval == ZIO_COMPRESS_LZ4 ||
3674                             intval == ZIO_COMPRESS_LZ4HC) {
```



```
3675                                 zfeature_info_t *feature =
3676                                     &spa_feature_table[
3677                                     SPA_FEATURE_LZ4_COMPRESS];
3678                                 spa_t *spa;
3680                                 if ((err = spa_open(dsname, &spa, FTAG)) != 0)
3681                                         return (err);
3683                                 if (!spa_feature_is_enabled(spa, feature)) {
3684                                         spa_close(spa, FTAG);
3685                                         return (ENOTSUP);
3686                                 }
3687                                 spa_close(spa, FTAG);
3688                         }
3690                         /*
3691                          * If this is a bootable dataset then
3692                          * verify that the compression algorithm
3693                          * is supported for booting. We must return
3694                          * something other than ENOTSUP since it
3695                          * implies a downrev pool version.
3696                          */
3697                         if (zfs_is_bootfs(dsname) &&
3698                             !BOOTFS_COMPRESS_VALID(intval)) {
3699                                 return (ERANGE);
3700                         }
3701                 }
3702                 break;
3704         case ZFS_PROP_COPIES:
3705                 if (zfs_earlier_version(dsname, SPA_VERSION_DITTO_BLOCKS))
3706                         return (ENOTSUP);
3707                 break;
3709         case ZFS_PROP_DEDUP:
3710                 if (zfs_earlier_version(dsname, SPA_VERSION_DEDUP))
3711                         return (ENOTSUP);
3712                 break;
3714         case ZFS_PROP_SHARESMB:
3715                 if (zpl_earlier_version(dsname, ZPL_VERSION_FUID))
3716                         return (ENOTSUP);
3717                 break;
3719         case ZFS_PROP_ACLINHERIT:
3720                 if (nvpair_type(pair) == DATA_TYPE_UINT64 &&
3721                     nvpair_value_uint64(pair, &intval) == 0) {
3722                         if (intval == ZFS_ACL_PASSTHROUGH_X &&
3723                             zfs_earlier_version(dsname,
3724                             SPA_VERSION_PASSTHROUGH_X))
3725                                 return (ENOTSUP);
3726                 }
3727                 break;
3728         }
3730         return (zfs_secpolicy_setprop(dsname, prop, pair, CRED()));
3731 }
3733 /*
3734  * Activates a feature on a pool in response to a property setting. This
3735  * creates a new sync task which modifies the pool to reflect the feature
3736  * as being active.
3737  */
3738 static int
3739 zfs_prop_activate_feature(dsl_pool_t *dp, zfeature_info_t *feature)
3740 {
```

```
3741         int err;

3743         /* EBUSY here indicates that the feature is already active */
3744         err = dsl_sync_task_do(dp, zfs_prop_activate_feature_check,
3745             zfs_prop_activate_feature_sync, dp->dp_spa, feature, 2);

3747         if (err != 0 && err != EBUSY)
3748                 return (err);
3749         else
3750                 return (0);
3751 }

3753 /*
3754  * Checks for a race condition to make sure we don't increment a feature flag
3755  * multiple times.
3756  */
3757 /*ARGSUSED*/
3758 static int
3759 zfs_prop_activate_feature_check(void *arg1, void *arg2, dmu_tx_t *tx)
3760 {
3761         spa_t *spa = arg1;
3762         zfeature_info_t *feature = arg2;

3764         if (!spa_feature_is_active(spa, feature))
3765                 return (0);
3766         else
3767                 return (EBUSY);
3768 }

3770 /*
3771  * The callback invoked on feature activation in the sync task caused by
3772  * zfs_prop_activate_feature.
3773  */
3774 static void
3775 zfs_prop_activate_feature_sync(void *arg1, void *arg2, dmu_tx_t *tx)
3776 {
3777         spa_t *spa = arg1;
3778         zfeature_info_t *feature = arg2;

3780         spa_feature_incr(spa, feature, tx);
3781 }

3783 /*
3784  * Removes properties from the given props list that fail permission checks
3785  * needed to clear them and to restore them in case of a receive error. For each
3786  * property, make sure we have both set and inherit permissions.
3787  *
3788  * Returns the first error encountered if any permission checks fail. If the
3789  * caller provides a non-NULL errlist, it also gives the complete list of names
3790  * of all the properties that failed a permission check along with the
3791  * corresponding error numbers. The caller is responsible for freeing the
3792  * returned errlist.
3793  *
3794  * If every property checks out successfully, zero is returned and the list
3795  * pointed at by errlist is NULL.
3796  */
3797 static int
3798 zfs_check_clearable(char *dataset, nvlist_t *props, nvlist_t **errlist)
3799 {
3800         zfs_cmd_t *zc;
3801         nvpair_t *pair, *next_pair;
3802         nvlist_t *errors;
3803         int err, rv = 0;

3805         if (props == NULL)
3806                 return (0);
```



```
3808         VERIFY(nvlist_alloc(&errors, NV_UNIQUE_NAME, KM_SLEEP) == 0);

3810         zc = kmem_alloc(sizeof (zfs_cmd_t), KM_SLEEP);
3811         (void) strcpy(zc->zc_name, dataset);
3812         pair = nvlist_next_nvpair(props, NULL);
3813         while (pair != NULL) {
3814                 next_pair = nvlist_next_nvpair(props, pair);

3816                 (void) strcpy(zc->zc_value, nvpair_name(pair));
3817                 if ((err = zfs_check_settable(dataset, pair, CRED())) != 0 ||
3818                     (err = zfs_secpolicy_inherit_prop(zc, NULL, CRED())) != 0) {
3819                         VERIFY(nvlist_remove_nvpair(props, pair) == 0);
3820                         VERIFY(nvlist_add_int32(errors,
3821                             zc->zc_value, err) == 0);
3822                 }
3823                 pair = next_pair;
3824         }
3825         kmem_free(zc, sizeof (zfs_cmd_t));

3827         if ((pair = nvlist_next_nvpair(errors, NULL)) == NULL) {
3828                 nvlist_free(errors);
3829                 errors = NULL;
3830         } else {
3831                 VERIFY(nvpair_value_int32(pair, &rv) == 0);
3832         }

3834         if (errlist == NULL)
3835                 nvlist_free(errors);
3836         else
3837                 *errlist = errors;

3839         return (rv);
3840 }
_____unchanged_portion_omitted_
```

****
3953 Tue Jan 15 15:38:25 2013
new/usr/src/uts/common/fs/zfs/zio_compress.c
%B
****

```
/*
 * CDDL HEADER START
 *
 * The contents of this file are subject to the terms of the
 * Common Development and Distribution License (the "License").
 * You may not use this file except in compliance with the License.
 *
 * You can obtain a copy of the license at usr/src/OPENSOLARIS.LICENSE
 * or http://www.opensolaris.org/os/licensing.
 * See the License for the specific language governing permissions
 * and limitations under the License.
 *
 * When distributing Covered Code, include this CDDL HEADER in each
 * file and include the License file at usr/src/OPENSOLARIS.LICENSE.
 * If applicable, add the following below this CDDL HEADER, with the
 * fields enclosed by brackets "[]" replaced with your own identifying
 * information: Portions Copyright [yyyy] [name of copyright owner]
 *
 * CDDL HEADER END
 */

/*
 * Copyright 2009 Sun Microsystems, Inc.  All rights reserved.
 * Use is subject to license terms.
 */
/*
 * Copyright (c) 2013 by Saso Kiselkov. All rights reserved.
 */

#include <sys/zfs_context.h>
#include <sys/compress.h>
#include <sys/spa.h>
#include <sys/zio.h>
#include <sys/zio_compress.h>

/*
 * Compression vectors.
 */

zio_compress_info_t zio_compress_table[ZIO_COMPRESS_FUNCTIONS] = {
        {NULL,                  NULL,                   0,      "inherit"},
        {NULL,                  NULL,                   0,      "on"},
        {NULL,                  NULL,                   0,      "uncompressed"},
        {lzjb_compress,         lzjb_decompress,        0,      "lzjb"},
        {NULL,                  NULL,                   0,      "empty"},
        {gzip_compress,         gzip_decompress,        1,      "gzip-1"},
        {gzip_compress,         gzip_decompress,        2,      "gzip-2"},
        {gzip_compress,         gzip_decompress,        3,      "gzip-3"},
        {gzip_compress,         gzip_decompress,        4,      "gzip-4"},
        {gzip_compress,         gzip_decompress,        5,      "gzip-5"},
        {gzip_compress,         gzip_decompress,        6,      "gzip-6"},
        {gzip_compress,         gzip_decompress,        7,      "gzip-7"},
        {gzip_compress,         gzip_decompress,        8,      "gzip-8"},
        {gzip_compress,         gzip_decompress,        9,      "gzip-9"},
        {zle_compress,          zle_decompress,         64,     "zle"},
        {lz4_compress,          lz4_decompress,         0,      "lz4"},
        {lz4_compress,          lz4_decompress,         1,      "lz4hc"},
};
_____unchanged_portion_omitted_
```