```
**********************************************************
   10340 Thu Jan 17 14:38:03 2019
new/usr/src/lib/libreparse/common/fs_reparse_lib.c
10106 libreparse needs smatch fixes
**********************************************************
/*
 * 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) 2018, Joyent, Inc.
 */

#include <stdio.h>
#include <stdlib.h>
#include <unistd.h>
#include <strings.h>
#include <string.h>
#include <dirent.h>
#include <sys/types.h>
#include <sys/stat.h>
#include <sys/param.h>
#include <sys/errno.h>
#include <limits.h>
#include <libnvpair.h>
#include <dlfcn.h>
#include <libintl.h>
#include <sys/systeminfo.h>
#include <sys/fs_reparse.h>
#include "rp_plugin.h"

#define MAXISALEN       257     /* based on sysinfo(2) man page */

static rp_proto_handle_t rp_proto_handle;
static rp_proto_plugin_t *rp_proto_list;

int rp_plugin_init(void);
static void proto_plugin_fini(void);
static rp_plugin_ops_t *rp_find_protocol(const char *svctype);

extern int errno;
static int rp_plugin_inited = 0;

/*
```

```
 * reparse_create()
 *
 * Create a symlink at the specified 'path' as a reparse point.
 * This function will fail if path refers to an existing file system
 * object or an object named string already exists at the given path.
 *
 * return 0 if ok else return error code.
 */
int
reparse_create(const char *path, const char *data)
{
        int err;
        struct stat sbuf;

        if (path == NULL || data == NULL)
                return (EINVAL);

        if ((err = reparse_validate(data)) != 0)
                return (err);

        /* check if object exists */
        if (lstat(path, &sbuf) == 0)
                return (EEXIST);

        return (symlink(data, path) ? errno : 0);
}
______unchanged_portion_omitted_

static void
proto_plugin_fini()
{
        rp_proto_plugin_t *p;

        /*
         * Protocols may call this framework during _fini
         */
        for (p = rp_proto_list; p != NULL; p = p->plugin_next) {
                if (p->plugin_ops->rpo_fini)
                        (void) p->plugin_ops->rpo_fini();
                        p->plugin_ops->rpo_fini();
        }
        while ((p = rp_proto_list) != NULL) {
                rp_proto_list = p->plugin_next;
                if (p->plugin_handle != NULL)
                        (void) dlclose(p->plugin_handle);
                free(p);
        }

        if (rp_proto_handle.rp_ops != NULL) {
                free(rp_proto_handle.rp_ops);
                rp_proto_handle.rp_ops = NULL;
        }
        rp_proto_handle.rp_num_proto = 0;
}
______unchanged_portion_omitted_
```