

```

*****
5058 Thu Jan 24 10:01:08 2019
new/usr/src/cmd/policykit/polkit-is-privileged.c
10136 smatch fix for policykit
*****
1 /*****
2  * CVSID: $Id$
3  *
4  * polkit-is-privileged.c : Determine if a user has privileges
5  *
6  * Copyright (C) 2006 David Zeuthen, <david@fubar.dk>
7  *
8  * This program is free software; you can redistribute it and/or modify
9  * it under the terms of the GNU General Public License as published by
10 * the Free Software Foundation; either version 2 of the License, or
11 * (at your option) any later version.
12 *
13 * This program is distributed in the hope that it will be useful,
14 * but WITHOUT ANY WARRANTY; without even the implied warranty of
15 * MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the
16 * GNU General Public License for more details.
17 *
18 * You should have received a copy of the GNU General Public License
19 * along with this program; if not, write to the Free Software
20 * Foundation, Inc., 51 Franklin St, Fifth Floor, Boston, MA 02110-1301 USA
21 *
22 *****/

24 /*
25  * Copyright (c) 2018, Joyent, Inc.
26  */

28 #ifdef HAVE_CONFIG_H
29 # include <config.h>
30 #endif

32 #include <stdio.h>
33 #include <stdlib.h>
34 #include <getopt.h>
35 #include <dbus/dbus.h>

37 #include <libpolkit/libpolkit.h>

39 static void
40 usage (int argc, char *argv[])
41 {
42     fprintf (stderr, "polkit-is-privileged version " PACKAGE_VERSION "\n");

44     fprintf (stderr,
45             "\n"
46             "usage : %s -u <uid> -p <privilege> [-r <resource>]\n"
47             "          [-s <system-bus-connection-name>]", argv[0]);

48     fprintf (stderr,
49             "\n"
50             "Options:\n"
51             "  -u, --user           Username or user id\n"
52             "  -s, --system-bus-unique-name Unique system bus connection
53             "  -r, --resource       Resource\n"
54             "  -p, --privilege      Privilege to test for\n"
55             "  -h, --help           Show this information and ex
56             "  -v, --verbose        Verbose operation\n"
57             "  -V, --version        Print version number\n"
58             "\n"
59             "Queries system policy whether a given user is allowed for a gi
60             "privilege for a given resource. The resource may be omitted.\n"
61             "\n");

```

```

62 }

64 int
65 main (int argc, char *argv[])
66 {
67     int rc;
68     char *user = NULL;
69     char *privilege = NULL;
70     char *resource = NULL;
71     char *system_bus_unique_name = NULL;
72     static const struct option long_options[] = {
73         {"user", required_argument, NULL, 'u'},
74         {"system-bus-unique-name", required_argument, NULL, 's'},
75         {"resource", required_argument, NULL, 'r'},
76         {"privilege", required_argument, NULL, 'p'},
77         {"help", no_argument, NULL, 'h'},
78         {"verbose", no_argument, NULL, 'v'},
79         {"version", no_argument, NULL, 'V'},
80         {NULL, 0, NULL, 0}
81     };
82     LibPolKitContext *ctx = NULL;
83     gboolean is_allowed;
84     gboolean is_temporary;
85     LibPolKitResult result;
86     gboolean is_verbose = FALSE;
87     DBusError error;
88     DBusConnection *connection = NULL;

90     rc = 1;

91     while (TRUE) {
92         int c;

93         c = getopt_long (argc, argv, "u:r:p:s:hVv", long_options, NULL);

94         if (c == -1)
95             break;

97         switch (c) {
98             case 's':
99                 system_bus_unique_name = g_strdup (optarg);
100                 break;
101             case 'u':
102                 user = g_strdup (optarg);
103                 break;
104             case 'r':
105                 resource = g_strdup (optarg);
106                 break;
107             case 'p':
108                 privilege = g_strdup (optarg);
109                 break;
110             case 'v':
111                 is_verbose = TRUE;
112                 break;
113             case 'h':
114                 usage (argc, argv);
115                 rc = 0;
116                 goto out;
117             case 'V':
118                 printf ("polkit-is-privileged version " PACKAGE_VERSION
119                 "\n");
120                 rc = 0;
121                 goto out;
122             case 'V':
123                 printf ("polkit-is-privileged version " PACKAGE_VERSION
124                 "\n");
125                 rc = 0;
126                 goto out;
127             default:
128                 usage (argc, argv);
129                 rc = 1;
130                 goto out;
131         }
132     }

133     if (system_bus_unique_name)
134         connection = dbus_connection_get_for_system_bus_name (system_bus_unique_name);
135     else
136         connection = dbus_connection_get_for_system_bus ();
137     if (!connection)
138         goto out;

139     ctx = libpolkit_context_new ();
140     if (!ctx)
141         goto out;

142     result = libpolkit_context_test_for_privilege (ctx, user, privilege, resource, connection, is_temporary);
143     is_allowed = libpolkit_result_is_allowed (&result);

144     if (is_verbose)
145         printf ("polkit-is-privileged: %s\n", is_allowed ? "allowed" : "not allowed");

146     rc = is_allowed ? 0 : 1;

147     out:
148     if (connection)
149         dbus_connection_close (connection);
150     if (ctx)
151         libpolkit_context_free (ctx);
152     return rc;

```

```

128             rc = 0;
129             goto out;
130
131         default:
132             usage (argc, argv);
133             goto out;
134     }
135 }
136
137 if (user == NULL || privilege == NULL) {
138     usage (argc, argv);
139     return 1;
140 }
141
142 if (is_verbose) {
143     printf ("user      = '%s'\n", user);
144     printf ("privilege = '%s'\n", privilege);
145     if (resource != NULL)
146         printf ("resource = '%s'\n", resource);
147 }
148
149 #ifdef POLKITD_ENABLED
150     dbus_error_init (&error);
151     connection = dbus_bus_get (DBUS_BUS_SYSTEM, &error);
152     if (connection == NULL) {
153         g_warning ("Cannot connect to system message bus");
154         return 1;
155     }
156 #endif /* POLKITD_ENABLED */
157
158     ctx = libpolkit_new_context (connection);
159     if (ctx == NULL) {
160         g_warning ("Cannot get libpolkit context");
161         goto out;
162     }
163
164     result = libpolkit_is_uid_allowed_for_privilege (ctx,
165                                                     system_bus_unique_name,
166                                                     user,
167                                                     privilege,
168                                                     resource,
169                                                     &is_allowed,
170                                                     &is_temporary,
171                                                     NULL);
172     switch (result) {
173     case LIBPOLKIT_RESULT_OK:
174         rc = is_allowed ? 0 : 1;
175         break;
176
177     case LIBPOLKIT_RESULT_ERROR:
178         g_warning ("Error determining whether user is privileged.");
179         break;
180
181     case LIBPOLKIT_RESULT_INVALID_CONTEXT:
182         g_print ("Invalid context.\n");
183         goto out;
184
185     case LIBPOLKIT_RESULT_NOT_PRIVILEGED:
186         g_print ("Not privileged.\n");
187         goto out;
188
189     case LIBPOLKIT_RESULT_NO_SUCH_PRIVILEGE:
190         g_print ("No such privilege '%s'.\n", privilege);
191         goto out;
192
193     case LIBPOLKIT_RESULT_NO_SUCH_USER:

```

```

194         g_print ("No such user '%s'.\n", user);
195         goto out;
196     }
197
198     if (is_verbose) {
199         printf ("result %d\n", result);
200         printf ("is_allowed %d\n", is_allowed);
201     }
202
203 out:
204     if (ctx != NULL)
205         (void) libpolkit_free_context (ctx);
206         libpolkit_free_context (ctx);
207
208     return rc;
209 }

```

unchanged\_portion\_omitted\_