owntone / **owntone-server** Public

Code | Issues 107 | Pull requests 6 | Actions | Wiki | Security and quality

⚠ This commit does not belong to any branch on this repository, and may belong to a fork outside of the repository.

Commit dca9464

yhcho0405 authored on Mar 9 · 5 / 5 · Verified

```
[daap] Fix session list race during concurrent logins
```

1 parent d4784eb commit dca9464

1 file changed +100 -37 lines changed

Top

Filter files…

src

httpd_daap.c

1 file changed +100 -37 lines changed

Search within code

src/httpd_daap.c

```
@@ -121,6 +121,7 @@ static char *default_meta_group =
"dmap.itemname,dmap.persistentid,daap.songalbu
121 121
122 122   /* DAAP session tracking */
123 123   static struct daap_session *daap_sessions;
    124 + static pthread_mutex_t daap_session_lck;
124 125
125 126   /* Update requests */
126 127   static int current_rev;
@@ -137,7 +138,7 @@ daap_session_free(struct daap_session *s)
```

```
137  138    }
138  139
139  140    static void
140       - daap_session_remove(struct daap_session *s)
     141  + daap_session_remove_locked(struct daap_session *s)
141  142    {
142  143      struct daap_session *ptr;
143  144      struct daap_session *prev;
          @@ -166,7 +167,7 @@ daap_session_remove(struct daap_session *s)
166  167    }
167  168
168  169    static struct daap_session *
169       - daap_session_get(uint32_t id)
     170  + daap_session_get_locked(uint32_t id)
170  171    {
171  172      struct daap_session *s;
172  173
          @@ -179,11 +180,66 @@ daap_session_get(uint32_t id)
179  180      return NULL;
180  181    }
181  182
     183  + static bool
     184  + daap_session_copy(struct daap_session *dst, uint32_t id)
     185  + {
     186  +   struct daap_session *s;
     187  +   bool is_found = false;
     188  +
     189  +   CHECK_ERR(L_DAAP, pthread_mutex_lock(&daap_session_lck));
     190  +
     191  +   s = daap_session_get_locked(id);
     192  +   if (s)
     193  +     {
     194  +       *dst = *s;
     195  +       dst->next = NULL;
     196  +       is_found = true;
     197  +     }
     198  +
     199  +   CHECK_ERR(L_DAAP, pthread_mutex_unlock(&daap_session_lck));
     200  +
```

```
201 +   return is_found;
202 + }
203 +
204 + static bool
205 + daap_session_touch(uint32_t id)
206 + {
207 +   struct daap_session *s;
208 +   bool is_found = false;
209 +
210 +   CHECK_ERR(L_DAAP, pthread_mutex_lock(&daap_session_lck));
211 +
212 +   s = daap_session_get_locked(id);
213 +   if (s)
214 +     {
215 +       s->mtime = time(NULL);
216 +       is_found = true;
217 +     }
218 +
219 +   CHECK_ERR(L_DAAP, pthread_mutex_unlock(&daap_session_lck));
220 +
221 +   return is_found;
222 + }
223 +
224 + static void
225 + daap_session_remove(uint32_t id)
226 + {
227 +   struct daap_session *s;
228 +
229 +   CHECK_ERR(L_DAAP, pthread_mutex_lock(&daap_session_lck));
230 +
231 +   s = daap_session_get_locked(id);
232 +   if (s)
233 +     daap_session_remove_locked(s);
234 +
235 +   CHECK_ERR(L_DAAP, pthread_mutex_unlock(&daap_session_lck));
236 + }
237 +
182 238   /* Removes stale sessions and also drops the oldest sessions if
      DAAP_SESSION_MAX
183 239    * will otherwise be exceeded
```

```
184 240      */
185 241     static void
186     -   daap_session_cleanup(void)
    242 +   daap_session_cleanup_locked(void)
187 243     {
188 244       struct daap_session *s;
189 245       struct daap_session *next;
        @@ -202,27 +258,30 @@ daap_session_cleanup(void)
202 258         {
203 259           DPRINTF(E_LOG, L_DAAP, "Cleaning up DAAP session (id %" PRIu32 ")\n",
                s->id);
204 260
205     -         daap_session_remove(s);
    261 +         daap_session_remove_locked(s);
206 262         }
207 263         }
208 264     }
209 265
210     - static struct daap_session *
211     - daap_session_add(bool is_remote, uint32_t request_session_id)
    266 + static int
    267 + daap_session_add(uint32_t *session_id, bool is_remote, uint32_t
                request_session_id)
212 268     {
213 269       struct daap_session *s;
214 270
215     -   daap_session_cleanup();
216     -
217 271       CHECK_NULL(L_DAAP, s = calloc(1, sizeof(struct daap_session)));
218 272
    273 +   CHECK_ERR(L_DAAP, pthread_mutex_lock(&daap_session_lck));
    274 +
    275 +   daap_session_cleanup_locked();
    276 +
219 277       if (request_session_id)
220 278         {
221     -         if (daap_session_get(request_session_id))
    279 +         if (daap_session_get_locked(request_session_id))
222 280         {
```

```
223  281          DPRINTF(E_LOG, L_DAAP, "Session id requested in login (%d) is not
                  available\n", request_session_id);
     282  +       CHECK_ERR(L_DAAP, pthread_mutex_unlock(&daap_session_lck));
224  283          free(s);
225       -       return NULL;
     284  +       return -1;
226  285        }
227  286
228  287          s->id = request_session_id;
                  @@ -233,7 +292,7 @@ daap_session_add(bool is_remote, uint32_t
                  request_session_id)
233  292          if (s->id < 100)
234  293        s->id += 100;
235  294        }
236       -   while (daap_session_get(s->id) != NULL);
     295  +   while (daap_session_get_locked(s->id) != NULL);
237  296
238  297      s->mtime = time(NULL);
239  298
                  @@ -244,7 +303,11 @@ daap_session_add(bool is_remote, uint32_t
                  request_session_id)
244  303
245  304      daap_sessions = s;
246  305
247       -   return s;
     306  +   *session_id = s->id;
     307  +
     308  +   CHECK_ERR(L_DAAP, pthread_mutex_unlock(&daap_session_lck));
     309  +
     310  +   return 0;
248  311    }
249  312
250  313    /* ---------------------- UPDATE REQUESTS HANDLERS -------------------------
                  */
                  @@ -731,8 +794,11 @@ daap_request_authorize(struct httpd_request *hreq)
731  794      session = hreq->extra_data;
732  795      if (session && session->id != 0)
733  796        {
734       -       session->mtime = time(NULL);
```

```
735       -       return 0;
          797 +   if (daap_session_touch(session->id))
          798 +     {
          799 +       session->mtime = time(NULL);
          800 +       return 0;
          801 +     }
736       802       }
737       803
738       804     passwd = cfg_getstr(cfg_getsec(cfg, "library"), "password");
                  @@ -904,8 +970,8 @@ static enum daap_reply_result
904       970     daap_reply_login(struct httpd_request *hreq)
905       971     {
906       972       struct daap_session *dummy = hreq->extra_data;
907       -       struct daap_session *session;
908       973       const char *param;
          974 +   uint32_t session_id;
909       975       uint32_t request_session_id = 0;
910       976       int ret;
911       977
                  @@ -919,8 +985,8 @@ daap_reply_login(struct httpd_request *hreq)
919       985         DPRINTF(E_LOG, L_DAAP, "Login request where request-session-id is not an
                  integer\n");
920       986         }
921       987
922       -       session = daap_session_add(dummy->is_remote, request_session_id);
923       -       if (!session)
          988 +   ret = daap_session_add(&session_id, dummy->is_remote, request_session_id);
          989 +   if (ret < 0)
924       990         {
925       991           dmap_error_make(hreq->out_body, "mlog", "Could not start session");
926       992           return DAAP_REPLY_ERROR;
                  @@ -930,20 +996,22 @@ daap_reply_login(struct httpd_request *hreq)
930       996
931       997       dmap_add_container(hreq->out_body, "mlog", 24);
932       998       dmap_add_int(hreq->out_body, "mstt", 200);          /* 12 */
933       -       dmap_add_int(hreq->out_body, "mlid", session->id);  /* 12 */
          999 +   dmap_add_int(hreq->out_body, "mlid", session_id);   /* 12 */
934      1000
935      1001       return DAAP_REPLY_OK;
```

```
936  1002    }
937  1003
938  1004    static enum daap_reply_result
939  1005    daap_reply_logout(struct httpd_request *hreq)
940  1006    {
941       -    if (!hreq->extra_data)
     1007  +    struct daap_session *session = hreq->extra_data;
     1008  +
     1009  +    if (!session || session->id == 0)
942  1010        return DAAP_REPLY_FORBIDDEN;
943  1011
944       -    daap_session_remove(hreq->extra_data);
     1012  +    daap_session_remove(session->id);
945  1013
946       -    hreq->extra_data = NULL;
     1014  +    memset(session, 0, sizeof(struct daap_session));
947  1015
948  1016      return DAAP_REPLY_LOGOUT;
949  1017    }
          @@ -2213,22 +2281,20 @@ daap_request(struct httpd_request *hreq)
2213 2281
2214 2282      // Check if we have a session and point hreq->extra_data to it
2215 2283      param = httpd_query_value_find(hreq->query, "session-id");
     2284  +    memset(&session, 0, sizeof(struct daap_session));
     2285  +    session.is_remote = (httpd_query_value_find(hreq->query, "pairing-guid") !=
          NULL);
     2286  +
2216 2287      if (param)
2217 2288        {
2218 2289          ret = safe_atou32(param, &id);
2219 2290          if (ret < 0)
2220 2291        DPRINTF(E_LOG, L_DAAP, "Ignoring non-numeric session id in DAAP request:
          '%s'\n", hreq->uri);
2221 2292          else
2222       -      hreq->extra_data = daap_session_get(id);
     2293  +      daap_session_copy(&session, id);
2223 2294        }
2224 2295
2225       -    // Create a dummy session to pass is_remote to the handler
```

```
2226      -   if (!hreq->extra_data)
2227      -     {
2228      -       memset(&session, 0, sizeof(struct daap_session));
2229      -       session.is_remote = (httpd_query_value_find(hreq->query, "pairing-
                guid") != NULL);
2230      -       hreq->extra_data = &session;
2231      -     }
          2296  +   // Handlers only need a request-local session snapshot
          2297  +   hreq->extra_data = &session;
2232      2298
2233      2299      if (strcmp(hreq->path, "/server-info") != 0 && strcmp(hreq->path,
                "/content-codes") != 0)
2234      2300        {
                  @@ -2274,14 +2340,7 @@ daap_request(struct httpd_request *hreq)
2274      2340    int
2275      2341    daap_session_is_valid(uint32_t id)
2276      2342    {
2277      -   struct daap_session *session;
2278      -
2279      -   session = daap_session_get(id);
2280      -
2281      -   if (session)
2282      -     session->mtime = time(NULL);
2283      -
2284      -   return session ? 1 : 0;
          2343  +   return daap_session_touch(id) ? 1 : 0;
2285      2344    }
2286      2345
2287      2346    // Thread: Cache
                  @@ -2335,6 +2394,7 @@ daap_init(void)
2335      2394    {
2336      2395      srand((unsigned)time(NULL));
2337      2396      current_rev = 2;
          2397  +   CHECK_ERR(L_DAAP, mutex_init(&daap_session_lck));
2338      2398
2339      2399      return 0;
2340      2400    }
                  @@ -2345,11 +2405,14 @@ daap_deinit(void)
```

```
2345  2405      struct daap_session *s;
2346  2406      struct daap_update_request *ur;
2347  2407
      2408  +   CHECK_ERR(L_DAAP, pthread_mutex_lock(&daap_session_lck));
2348  2409      for (s = daap_sessions; daap_sessions; s = daap_sessions)
2349  2410        {
2350  2411          daap_sessions = s->next;
2351  2412          daap_session_free(s);
2352  2413        }
      2414  +   CHECK_ERR(L_DAAP, pthread_mutex_unlock(&daap_session_lck));
      2415  +   CHECK_ERR(L_DAAP, pthread_mutex_destroy(&daap_session_lck));
2353  2416
2354  2417      for (ur = update_requests; update_requests; ur = update_requests)
2355  2418        {
```

Comments 0

Please sign in to comment.