

```
new/usr/src/head/unistd.h  
*****  
26278 Sun Mar 29 19:44:23 2015  
new/usr/src/head/unistd.h  
5776 vfork and getwd should not be exposed under XPG7  
*****
```

```
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 2014 Garrett D'Amore <garrett@damore.org>  
24 * Copyright (c) 2013 Gary Mills  
25 *  
26 * Copyright (c) 1989, 2010, Oracle and/or its affiliates. All rights reserved.  
27 */  
  
29 /* Copyright (c) 1988 AT&T */  
30 /* All Rights Reserved */  
  
32 /* Copyright (c) 2013, OmniTI Computer Consulting, Inc. All rights reserved. */  
  
34 #ifndef _UNISTD_H  
35 #define _UNISTD_H  
  
37 #include <sys/feature_tests.h>  
  
39 #include <sys/types.h>  
40 #include <sys/unistd.h>  
  
42 #ifdef __cplusplus  
43 extern "C" {  
44 #endif  
  
46 /* Symbolic constants for the "access" routine: */  
47 #define R_OK 4 /* Test for Read permission */  
48 #define W_OK 2 /* Test for Write permission */  
49 #define X_OK 1 /* Test for eXecute permission */  
50 #define F_OK 0 /* Test for existence of File */  
  
52 #if !defined(_XOPEN_OR_POSIX) || defined(_XPG4_2) || defined(__EXTENSIONS__)  
53 #define F_ULOCK 0 /* Unlock a previously locked region */  
54 #define F_LOCK 1 /* Lock a region for exclusive use */  
55 #define F_TLOCK 2 /* Test and lock a region for exclusive use */  
56 #define F_TEST 3 /* Test a region for other processes locks */  
57 #endif /* !defined(_XOPEN_OR_POSIX) || defined(_XPG4_2)... */  
  
59 /* Symbolic constants for the "lseek" routine: */  
  
61 #ifndef SEEK_SET
```

```
1  
new/usr/src/head/unistd.h  
62 #define SEEK_SET 0 /* Set file pointer to "offset" */  
63 #endif  
  
65 #ifndef SEEK_CUR 1 /* Set file pointer to current plus "offset" */  
66 #define SEEK_CUR 1  
67 #endif  
  
69 #ifndef SEEK_END 2 /* Set file pointer to EOF plus "offset" */  
70 #define SEEK_END 2  
71 #endif  
  
73 #if !defined(_XOPEN_OR_POSIX) || defined(__EXTENSIONS__)  
74 #ifndef SEEK_DATA 3 /* Set file pointer to next data past offset */  
75 #define SEEK_DATA 3  
76 #endif  
  
78 #ifndef SEEK_HOLE 4 /* Set file pointer to next hole past offset */  
79 #define SEEK_HOLE 4  
80 #endif  
81 #endif /* !defined(_XOPEN_OR_POSIX) || defined(__EXTENSIONS__ */  
  
83 #if !defined(_XOPEN_OR_POSIX) || defined(__EXTENSIONS__)  
84 /* Path names: */  
85 #define GF_PATH "/etc/group" /* Path name of the "group" file */  
86 #define PF_PATH "/etc/passwd" /* Path name of the "passwd" file */  
87 #endif /* !defined(_XOPEN_OR_POSIX) || defined(__EXTENSIONS__ */  
  
89 /*  
90 * compile-time symbolic constants,  
91 * Support does not mean the feature is enabled.  
92 * Use pathconf/sysconf to obtain actual configuration value.  
93 */  
  
95 /* Values unchanged in UNIX 03 */  
96 #define _POSIX_ASYNC_IO 1  
97 #define _POSIX_JOB_CONTROL 1  
98 #define _POSIX_SAVED_IDS 1  
99 #define _POSIX_SYNC_IO 1  
  
101 /*  
102 * POSIX.1b compile-time symbolic constants.  
103 */  
104 #if defined(_XPG6)  
105 #define _POSIX_ASYNCNCHRONOUS_IO 200112L  
106 #define _POSIX_FSYNC 200112L  
107 #define _POSIX_MAPPED_FILES 200112L  
108 #define _POSIX_MEMLOCK 200112L  
109 #define _POSIX_MEMLOCK_RANGE 200112L  
110 #define _POSIX_MEMORY_PROTECTION 200112L  
111 #define _POSIX_MESSAGE_PASSING 200112L  
112 #define _POSIX_PRIORITY_SCHEDULING 200112L  
113 #define _POSIX_REALTIME_SIGNALS 200112L  
114 #define _POSIX_SEMAPHORES 200112L  
115 #define _POSIX_SHARED_MEMORY_OBJECTS 200112L  
116 #define _POSIX_SYNCHRONIZED_IO 200112L  
117 #else  
118 #define _POSIX_ASYNCNCHRONOUS_IO 1  
119 #define _POSIX_FSYNC 1  
120 #define _POSIX_MAPPED_FILES 1  
121 #define _POSIX_MEMLOCK 1  
122 #define _POSIX_MEMLOCK_RANGE 1  
123 #define _POSIX_MEMORY_PROTECTION 1  
124 #define _POSIX_MESSAGE_PASSING 1  
125 #define _POSIX_PRIORITY_SCHEDULING 1  
126 #define _POSIX_REALTIME_SIGNALS 1  
127 #define _POSIX_SEMAPHORES 1
```

```

128 #define _POSIX_SHARED_MEMORY_OBJECTS      1
129 #define _POSIX_SYNCHRONIZED_IO          1
130 #endifif

132 /*
133  * POSIX.1c compile-time symbolic constants.
134 */
135 #if defined(_XPG6)
136 #define _POSIX_THREAD_SAFE_FUNCTIONS      200112L
137 #define _POSIX_THREADS                  200112L
138 #define _POSIX_THREAD_ATTR_STACKADDR    200112L
139 #define _POSIX_THREAD_ATTR_STACKSIZE    200112L
140 #define _POSIX_THREAD_PROCESS_SHARED    200112L
141 #define _POSIX_THREAD_PRIORITY_SCHEDULING 200112L
142 #define _POSIX_TIMERS                  200112L
143 #else
144 #define _POSIX_THREAD_SAFE_FUNCTIONS      1
145 #define _POSIX_THREADS                  1
146 #define _POSIX_THREAD_ATTR_STACKADDR    1
147 #define _POSIX_THREAD_ATTR_STACKSIZE    1
148 #define _POSIX_THREAD_PROCESS_SHARED    1
149 #define _POSIX_THREAD_PRIORITY_SCHEDULING 1
150 #define _POSIX_TIMERS                  1
151 #endifif

153 /* New in UNIX 03 */
154 #define _POSIX_ADVISORY_INFO           200112L
155 #define _POSIX_BARRIERS                200112L
156 #define _POSIX_CLOCK_SELECTION         200112L
157 #define _POSIX_IPV6                  200112L
158 #define _POSIX_MONOTONIC_CLOCK        200112L
159 #define _POSIX_RAW_SOCKETS            200112L
160 #define _POSIX_READER_WRITER_LOCKS    200112L
161 #define _POSIX_SPAWN                 200112L
162 #define _POSIX_SPIN_LOCKS             200112L
163 #define _POSIX_TIMEOUTS              200112L

165 /*
166  * Support for the POSIX.1 mutex protocol attribute. For realtime applications
167  * which need mutexes to support priority inheritance/ceiling.
168 */
169 #if defined(_XPG6)
170 #define _POSIX_THREAD_PRIO_INHERIT    200112L
171 #define _POSIX_THREAD_PRIO_PROTECT    200112L
172 #else
173 #define _POSIX_THREAD_PRIO_INHERIT    1
174 #define _POSIX_THREAD_PRIO_PROTECT    1
175 #endifif

177 #ifndef _POSIX_VDISABLE
178 #define _POSIX_VDISABLE               0
179 #endifif

181 #ifndef NULL
182 #if defined(_LP64)
183 #define NULL    0L
184 #else
185 #define NULL    0
186 #endifif
187 #endifif

189 #define STDIN_FILENO     0
190 #define STDOUT_FILENO   1
191 #define STDERR_FILENO   2
193 /*

```

```

194  * Large File Summit-related announcement macros. The system supports both
195  * the additional and transitional Large File Summit interfaces. (The final
196  * two macros provide a finer granularity breakdown of _LFS64_LARGEFILE.)
197 */
198 #define _LFS_LARGEFILE           1
199 #define _LFS64_LARGEFILE         1
200 #define _LFS64_STUDIO           1
201 #define _LFS64_ASYNCNCHRONOUS_IO 1

203 /* large file compilation environment setup */
204 #if !defined(_LP64) && _FILE_OFFSET_BITS == 64
205 #ifdef __PRAGMA_REDEFINE_EXTNAME
206 #pragma redefine_extname ftruncate ftruncate64
207 #pragma redefine_extname lseek lseek64
208 #pragma redefine_extname pread pread64
209 #pragma redefine_extname pwrite pwrite64
210 #pragma redefine_extname truncate truncate64
211 #pragma redefine_extname lockf lockf64
212 #pragma redefine_extname tell tell64
213 #else /* __PRAGMA_REDEFINE_EXTNAME */
214 #define ftruncate ftruncate64
215 #define lseek lseek64
216 #define pread pread64
217 #define pwrite pwrite64
218 #define truncate truncate64
219 #define lockf lockf64
220 #define tell tell64
221 #endif /* __PRAGMA_REDEFINE_EXTNAME */
222 #endif /* !_LP64 && _FILE_OFFSET_BITS == 64 */

224 /* In the LP64 compilation environment, the APIs are already large file */
225 #if defined(_LP64) && defined(_LARGEFILE64_SOURCE)
226 #ifdef __PRAGMA_REDEFINE_EXTNAME
227 #pragma redefine_extname ftruncate64 ftruncate
228 #pragma redefine_extname lseek64 lseek
229 #pragma redefine_extname pread64 pread
230 #pragma redefine_extname pwrite64 pwrite
231 #pragma redefine_extname truncate64 truncate
232 #pragma redefine_extname lockf64 lockf
233 #pragma redefine_extname tell64 tell
234 #else /* __PRAGMA_REDEFINE_EXTNAME */
235 #define ftruncate64 ftruncate
236 #define lseek64 lseek
237 #define pread64 pread
238 #define pwrite64 pwrite
239 #define truncate64 truncate
240 #define lockf64 lockf
241 #define tell64 tell
242 #endif /* __PRAGMA_REDEFINE_EXTNAME */
243 #endif /* _LP64 && _LARGEFILE64_SOURCE */

245 extern int access(const char *, int);
246 #if !defined(__XOPEN_OR_POSIX) || defined(__EXTENSIONS__)
247 extern int acct(const char *);
248 #endifif /* !defined(__XOPEN_OR_POSIX) || defined(__EXTENSIONS__) */
249 extern unsigned alarm(unsigned);
250 /* Marked as LEGACY in SUSv2 and removed in SUSv3 */
251 #if !defined(__XOPEN_OR_POSIX) || ((defined(_XPG4_2) && !defined(_XPG6)) || \
252 defined(__EXTENSIONS__))
253 extern int brk(void *);
254 #endifif /* !defined(__XOPEN_OR_POSIX) || (defined(_XPG4_2)... */
255 extern int chdir(const char *);
256 extern int chown(const char *, uid_t, gid_t);
257 /* Marked as LEGACY in SUSv2 and removed in SUSv3 */
258 #if !defined(__POSIX_C_SOURCE) || ((defined(__XOPEN_SOURCE) && \
259 !defined(_XPG6)) || defined(__EXTENSIONS__))

```

```

260 extern int chroot(const char *);
261 #endif /* !defined(_POSIX_C_SOURCE) || defined(_XOPEN_SOURCE))... */
262 extern int close(int);
263 #if defined(_XPG4) || defined(__EXTENSIONS__)
264 extern size_t confstr(int, char *, size_t);
265 extern char *crypt(const char *, const char *);
266 #endif /* defined(_XPG4) || defined(__EXTENSIONS__) */
267 #if !defined(_POSIX_C_SOURCE) || defined(_XOPEN_SOURCE) || \
268     defined(__EXTENSIONS__)
269 extern char *ctermid(char *);
270 #endif /* (!defined(_POSIX_C_SOURCE) ... */
271 #if !defined(_XOPEN_OR_POSIX) || defined(_REENTRANT) || defined(__EXTENSIONS__)
272 extern char *ctermid_r(char *);
273 #endif /* !defined(_XOPEN_OR_POSIX) || defined(_REENTRANT) ... */
274 /* Marked as LEGACY in SUSv2 and removed in SUSv3 */
275 #if !defined(_XPG6) || defined(__EXTENSIONS__)
276 extern char *cuserid(char *);
277 #endif
278 extern int dup(int);
279 extern int dup2(int, int);
280 extern int dup3(int, int, int);
281 #if defined(_XPG4) || defined(__EXTENSIONS__)
282 extern void encrypt(char *, int);
283 #endif /* defined(_XPG4) || defined(__EXTENSIONS__) */
284 #if !defined(_XOPEN_OR_POSIX) || defined(__EXTENSIONS__)
285 extern void endusershell(void);
286 #endif /* !defined(_XOPEN_OR_POSIX) || defined(__EXTENSIONS__) */
287 extern int execl(const char *, const char *, ...);
288 extern int execle(const char *, const char *, ...);
289 extern int execlp(const char *, const char *, ...);
290 extern int execv(const char *, char *const *);
291 extern int execve(const char *, char *const *, char *const *);
292 extern int execvp(const char *, char *const *);
293 extern void _exit(int)
294     __NOTRETURN;
295 /*
296 * The following fattach prototype is duplicated in <stropts.h>. The
297 * duplication is necessitated by XPG4.2 which requires the prototype
298 * be defined in <stropts.h>.
299 */
300 #if !defined(_XOPEN_OR_POSIX) || defined(__EXTENSIONS__)
301 extern int fattach(int, const char *);
302 #endif /* !defined(_XOPEN_OR_POSIX) || defined(__EXTENSIONS__) */
303 #if !defined(_XOPEN_OR_POSIX) || defined(_XPG4_2) || defined(__EXTENSIONS__)
304 extern int fchdir(int);
305 extern int fchown(int, uid_t, gid_t);
306 #endif /* !defined(_XOPEN_OR_POSIX) || defined(_XPG4_2)... */
307 #if !defined(_XOPEN_OR_POSIX) || defined(__EXTENSIONS__)
308 extern int fcroot(int);
309 #endif /* !defined(_XOPEN_OR_POSIX) || defined(__EXTENSIONS__) */
310 #if !defined(_XOPEN_OR_POSIX) || (_POSIX_C_SOURCE > 2) || \
311     defined(__EXTENSIONS__)
312 extern int fdatasync(int);
313 #endif /* !defined(_XOPEN_OR_POSIX) || (_POSIX_C_SOURCE > 2)... */
314 /*
315 * The following fdetach prototype is duplicated in <stropts.h>. The
316 * duplication is necessitated by XPG4.2 which requires the prototype
317 * be defined in <stropts.h>.
318 */
319 #if !defined(_XOPEN_OR_POSIX) || defined(__EXTENSIONS__)
320 extern int fdetach(const char *);
321 #endif /* !defined(_XOPEN_OR_POSIX)... */
322 extern pid_t fork(void);
323 #if !defined(_XOPEN_OR_POSIX) || defined(__EXTENSIONS__)
324 extern pid_t fork1(void);
325 extern pid_t forkall(void);

```

```

326 #endif /* !defined(_XOPEN_OR_POSIX)... */
327 extern long fpathconf(int, int);
328 #if !defined(_POSIX_C_SOURCE) || (_POSIX_C_SOURCE > 2) || \
329     defined(__EXTENSIONS__)
330 extern int fsync(int);
331 #endif /* !defined(_POSIX_C_SOURCE) || (_POSIX_C_SOURCE > 2)... */
332 #if !defined(_XOPEN_OR_POSIX) || (_POSIX_C_SOURCE > 2) || \
333     (defined(_LARGEFILE_SOURCE) && _FILE_OFFSET_BITS == 64) || \
334     defined(__EXTENSIONS__)
335 extern int ftruncate(int, off_t);
336 #endif /* !defined(_XOPEN_OR_POSIX) || (_POSIX_C_SOURCE > 2)... */
337 extern char *getcwd(char *, size_t);
338 #if !defined(_XOPEN_OR_POSIX) || (defined(_XPG4_2) && !defined(_XPG6)) || \
339     defined(__EXTENSIONS__)
340 extern int getdtablesize(void);
341 #endif
342 extern gid_t getegid(void);
343 extern uid_t geteuid(void);
344 extern gid_t getgid(void);
345 extern int getgroups(int, gid_t *);
346 #if !defined(_XOPEN_OR_POSIX) || defined(_XPG4_2) || defined(__EXTENSIONS__)
347 extern long gethostid(void);
348 #endif
349 #if defined(_XPG4_2)
350 extern int gethostname(char *, size_t);
351 #elif !defined(_XOPEN_OR_POSIX) || defined(__EXTENSIONS__)
352 extern int gethostname(char *, int);
353 #endif
355 #ifndef __GETLOGIN_DEFINED      /* Avoid duplicate in stdlib.h */
356 #define __GETLOGIN_DEFINED
357 #ifndef __USE_LEGACY_LOGINNAME
358 #ifdefined(_PRAGMA_REDEFINE_EXTNAME)
359 #pragma redefine_extname getlogin getloginx
360 #else /* __PRAGMA_REDEFINE_EXTNAME */
361 extern char *getloginx(void);
362 #define getlogin    getloginx
363 #endif /* __PRAGMA_REDEFINE_EXTNAME */
364 #endif /* __USE_LEGACY_LOGINNAME */
365 extern char *getlogin(void);
366 #endif /* __GETLOGIN_DEFINED */
368 #if defined(_XPG4) || defined(__EXTENSIONS__)
369 extern int getopt(int, char *const *, const char *);
370 extern char *optarg;
371 extern int opterr, optind, optarg;
372 /* Marked as LEGACY in SUSv2 and removed in SUSv3 */
373 #if !defined(_XPG6) || defined(__EXTENSIONS__)
374 extern char *getpass(const char *);
375 #endif
376 #endif /* defined(_XPG4) || defined(__EXTENSIONS__) */
377 #if !defined(_XOPEN_OR_POSIX) || defined(_XPG4_2) || defined(__EXTENSIONS__)
378 /* Marked as LEGACY in SUSv2 and removed in SUSv3 */
379 #if !defined(_XPG6) || defined(__EXTENSIONS__)
380 extern int getpagesize(void);
381 #endif
382 extern pid_t getpgid(pid_t);
383 #endif /* !defined(_XOPEN_OR_POSIX) || defined(_XPG4_2)... */
384 extern pid_t getpid(void);
385 extern pid_t getppid(void);
386 extern pid_t getpgrp(void);
388 #if !defined(_XOPEN_OR_POSIX) || defined(__EXTENSIONS__)
389 char *gettext(const char *, const char *);
390 #endif /* !defined(_XOPEN_OR_POSIX) || defined(__EXTENSIONS__) */
391 #if !defined(_XOPEN_OR_POSIX) || defined(_XPG4_2) || defined(__EXTENSIONS__)

```

```

392 extern pid_t getsid(pid_t);
393 #endif /* !defined(_XOPEN_OR_POSIX) || defined(_XPG4_2)... */
394 extern uid_t getuid(void);
395 #if !defined(_XOPEN_OR_POSIX) || defined(__EXTENSIONS__)
396 extern char *getusershell(void);
397 #endif /* !defined(_XOPEN_OR_POSIX) || defined(__EXTENSIONS...) */
398 #if !defined(_XOPEN_OR_POSIX) || defined(_XPG4_2) || defined(__EXTENSIONS__)
399 extern char *getwd(char *);
400 #endif /* !defined(_XOPEN_OR_POSIX) || defined(_XPG4_2)... */
401 /* * The following ioctl prototype is duplicated in <stropts.h>. The
402 * duplication is necessitated by XPG4.2 which requires the prototype
403 * be defined in <stropts.h>.
404 */
405 #if !defined(_XOPEN_OR_POSIX) || defined(__EXTENSIONS__)
406 extern int ioctl(int, int, ...);
407 extern int isexec(const char *, char *const *, char *const *);
408 extern int isetugid(void);
409 #endif
410 extern int isatty(int);
411 #if !defined(_XOPEN_OR_POSIX) || defined(_XPG4_2) || defined(__EXTENSIONS__)
412 extern int lchown(const char *, uid_t, gid_t);
413 #endif
414 extern int link(const char *, const char *);
415 #if !defined(_XOPEN_OR_POSIX) || defined(__EXTENSIONS__)
416 extern offset_t llseek(int, offset_t, int);
417 #endif
418 #if !defined(_XOPEN_OR_POSIX) || defined(_XPG4_2) || \
419     (defined(_LARGEFILE_SOURCE) && _FILE_OFFSET_BITS == 64) || \
420     defined(__EXTENSIONS__)
421 extern int lockf(int, int, off_t);
422 #endif
423 extern off_t lseek(int, off_t, int);
424 #if !defined(_POSIX_C_SOURCE) || defined(_XOPEN_SOURCE) || \
425     defined(__EXTENSIONS__)
426 extern int nice(int);
427 #endif /* !defined(_POSIX_C_SOURCE) || defined(_XOPEN_SOURCE)... */
428 #if !defined(_XOPEN_OR_POSIX) || defined(__EXTENSIONS__)
429 extern long pathconf(const char *, int);
430 extern int pause(void);
431 extern int pipe(int *);
432 extern int pipe2(int *, int);
433 #if !defined(_POSIX_C_SOURCE) || defined(_XPG5) || \
434     (defined(_LARGEFILE_SOURCE) && _FILE_OFFSET_BITS == 64) || \
435     defined(__EXTENSIONS__)
436 extern ssize_t pread(int, void *, size_t, off_t);
437 #endif
438 #if !defined(_XOPEN_OR_POSIX) || defined(__EXTENSIONS__)
439 extern void profil(unsigned short *, size_t, unsigned long, unsigned int);
440 #endif
441 /*
442 * pthread_atfork() is also declared in <pthread.h> as per SUSv3. The
443 * declarations are identical. A change to either one may also require
444 * appropriate namespace updates in order to avoid redeclaration
445 * warnings in the case where both prototypes are exposed via inclusion
446 * of both <pthread.h> and <unistd.h>.
447 */
448 #if !defined(_XOPEN_OR_POSIX) || \
449     ((-_POSIX_C_SOURCE > 2) && !defined(_XPG6)) || \
450     defined(__EXTENSIONS__)
451 extern int pthread_atfork(void (*) (void), void (*) (void), void (*) (void));
452 #endif /* !defined(_XOPEN_OR_POSIX) || ((-_POSIX_C_SOURCE > 2) ... */
453 #if !defined(_LP64) && \
454     (!defined(_XOPEN_OR_POSIX) || defined(__EXTENSIONS__))

```

```

455 extern int ptrace(int, pid_t, int, int);
456 #endif
457 #if !defined(_POSIX_C_SOURCE) || defined(_XPG5) || \
458     (defined(_LARGEFILE_SOURCE) && _FILE_OFFSET_BITS == 64) || \
459     defined(__EXTENSIONS__)
460 extern ssize_t pwrite(int, const void *, size_t, off_t);
461 #endif
462 #if !defined(_XOPEN_OR_POSIX) || defined(__EXTENSIONS__)
463 /* per RFC 3542; This is also defined in netdb.h */
464 extern int rcmd_af(char **, unsigned short, const char *, const char *,
465                     const char *, int *, int);
466 #endif
467 extern ssize_t read(int, void *, size_t);
468 #if !defined(_XOPEN_OR_POSIX) || \
469     defined(_XPG4_2) || defined(__EXTENSIONS__)
470 extern ssize_t readlink(const char *_RESTRICT_KYWD, char *_RESTRICT_KYWD,
471                       size_t);
472 #endif
473 #if !(defined(_XOPEN_OR_POSIX) || (defined(_XPG3) && !defined(_XPG4))) || \
474     defined(__EXTENSIONS__)
475 #if __cplusplus >= 199711L
476 namespace std {
477 #endif
478 extern int rename(const char *, const char *);
479 #if __cplusplus >= 199711L
480 } /* end of namespace std */

482 using std::rename;
483 #endif /* __cplusplus >= 199711L */
484 #endif /* !(defined(_XOPEN_OR_POSIX) || defined(__EXTENSIONS__)) */
485 #if !defined(_XOPEN_OR_POSIX) || defined(__EXTENSIONS__)
486 extern int resolvepath(const char *, char *, size_t);
487 /* per RFC 3542; This is also defined in netdb.h */
488 extern int rexec_af(char **, unsigned short, const char *, const char *,
489                      const char *, int *, int);
490 #endif /* !(defined(_XOPEN_OR_POSIX) || defined(__EXTENSIONS__)) */
491 extern int rmdir(const char *);
492 #if !defined(_XOPEN_OR_POSIX) || defined(__EXTENSIONS__)
493 /* per RFC 3542; This is also defined in netdb.h */
494 extern int rresvport_af(int *, int);
495 #endif

497 #if !defined(_XOPEN_OR_POSIX) || (defined(_XPG4_2) && !defined(_XPG6)) || \
498     defined(__EXTENSIONS__)
499 extern void *sbrk(intptr_t);
500 #endif /* !defined(_XOPEN_OR_POSIX) || (defined(_XPG4_2)... */
501 #if !defined(_XOPEN_OR_POSIX) || defined(_XPG6) || defined(__EXTENSIONS__)
502 extern int setegid(gid_t);
503 extern int seteuid(uid_t);
504 #endif /* !defined(_XOPEN_OR_POSIX) || defined(_XPG6) ... */
505 extern int setgid(gid_t);
506 #if !defined(_XOPEN_OR_POSIX) || defined(__EXTENSIONS__)
507 extern int setgroups(int, const gid_t *);
508 extern int sethostname(char *, int);
509 #endif /* !defined(_XOPEN_OR_POSIX) || defined(__EXTENSIONS__) */
510 extern int setpgid(pid_t, pid_t);
511 #if !defined(_XOPEN_OR_POSIX) || defined(_XPG4_2) || defined(__EXTENSIONS__)
512 extern pid_t setpgrp(void);
513 extern int setregid(gid_t, gid_t);
514 extern int setreuid(uid_t, uid_t);
515 #endif /* !defined(_XOPEN_OR_POSIX) || defined(_XPG4_2)... */
516 extern pid_t setsid(void);
517 extern int setuid(uid_t);
518 #if !defined(_XOPEN_OR_POSIX) || defined(__EXTENSIONS__)
519 extern void setusershell(void);
520 #endif /* !defined(_XOPEN_OR_POSIX) || defined(__EXTENSIONS__) */

```

```

521 extern unsigned sleep(unsigned);
522 #if !defined(_XOPEN_OR_POSIX) || defined(__EXTENSIONS__)
523 extern int stime(const time_t *);
524 #endif /* !defined(_XOPEN_OR_POSIX) || defined(__EXTENSIONS__) */
525 #if defined(_XPG4)
526 /* __EXTENSIONS__ makes the SVID Third Edition prototype in stdlib.h visible */
527 extern void swab(const void * _RESTRICT_KYWD, void * _RESTRICT_KYWD, ssize_t);
528 #endif /* defined(_XPG4) */
529 #if !defined(_XOPEN_OR_POSIX) || defined(_XPG4_2) || defined(__EXTENSIONS__)
530 extern int symlink(const char *, const char *);
531 extern void sync(void);
532 #endif /* !defined(_XOPEN_OR_POSIX) || defined(_XPG4_2) */
533 #if defined(_XPG5) && !defined(_XPG6)
534 #ifdef __PRAGMA_REDEFINE_EXTNAME
535 #pragma redefine_extname sysconf __sysconf_xpg5
536 #else /* __PRAGMA_REDEFINE_EXTNAME */
537 #define sysconf __sysconf_xpg5
538 #endif /* __PRAGMA_REDEFINE_EXTNAME */
539 #endif /* defined(_XPG5) && !defined(_XPG6) */
540 extern long sysconf(int);
541 extern pid_t tcgetpgrp(int);
542 extern int tcsetpgrp(int, pid_t);
543 #if !defined(_XOPEN_OR_POSIX) || \
544     (defined(__LARGEFILE_SOURCE) && __FILE_OFFSET_BITS == 64) || \
545     defined(__EXTENSIONS__)
546 extern off_t tell(int);
547 #endif /* !defined(_XOPEN_OR_POSIX)... */
548 #if !defined(_XOPEN_OR_POSIX) || defined(_XPG4_2) || \
549     (defined(__LARGEFILE_SOURCE) && __FILE_OFFSET_BITS == 64) || \
550     defined(__EXTENSIONS__)
551 extern int truncate(const char *, off_t);
552 #endif /* !defined(_XOPEN_OR_POSIX) || defined(_XPG4_2)... */
553 extern char *ttyname(int);
554 #if (defined(_XPG4_2) && !defined(_XPG7)) || !defined(__STRICT_SYMBOLS)
555 extern useconds_t ualarm(useconds_t, useconds_t);
556 #endif
557 extern int unlink(const char *);
558 #if (defined(_XPG4_2) && !defined(_XPG7)) || !defined(__STRICT_SYMBOLS)
559 extern char *getwd(char *);
560 extern int usleep(useconds_t);
561 extern pid_t vfork(void) __RETURNS_TWICE;
562 #pragma unknown_control_flow(vfork)
563 #endif
564 #if !defined(_XOPEN_OR_POSIX) || defined(_XPG4_2) || defined(__EXTENSIONS__)
565 extern pid_t vfork(void) __RETURNS_TWICE;
566 #endif /* !defined(_XOPEN_OR_POSIX) || defined(_XPG4_2)... */
567 #if !defined(_XOPEN_OR_POSIX) || defined(__EXTENSIONS__)
568 extern ssize_t write(int, const void *, size_t);
569 #if !defined(_XOPEN_OR_POSIX) || defined(__EXTENSIONS__)
570 extern void yield(void);
571 #endif /* !defined(_XOPEN_OR_POSIX) || defined(__EXTENSIONS__) */
572 #if !defined(_XOPEN_OR_POSIX) || defined(__ATFILE_SOURCE) || \
573     defined(__EXTENSIONS__)
574     /* !defined(_XPG7) */
575 extern int faccessat(int, const char *, int, int);
576 extern int fchownat(int, const char *, uid_t, gid_t, int);
577 extern int linkat(int, const char *, int, const char *, int);
578 extern ssize_t readlinkat(int, const char * _RESTRICT_KYWD,
579                         char * _RESTRICT_KYWD, size_t);
580 extern int renameat(int, const char *, int, const char *);
581 extern int symlinkat(const char *, int, const char *);
582 extern int unlinkat(int, const char *, int);
583 #endif /* !defined(_XOPEN_OR_POSIX) || defined(__ATFILE_SOURCE)... */

```

```

584 #if !defined(_XOPEN_OR_POSIX) || defined(__EXTENSIONS__)
585 extern int get_nprocs(void);
586 extern int get_nprocs_conf(void);
587 #endif /* !defined(_XOPEN_OR_POSIX) || defined(__EXTENSIONS__) */
588 /* transitional large file interface versions */
589 #if defined(__LARGEFILE64_SOURCE) && !((__FILE_OFFSET_BITS == 64) && \
590     !defined(__PRAGMA_REDEFINE_EXTNAME))
591     !defined(__PRAGMA_REDEFINE_EXTNAME))
592 extern int ftruncate64(int, off64_t);
593 extern off64_t lseek64(int, off64_t, int);
594 extern ssize_t pread64(int, void *, size_t, off64_t);
595 extern ssize_t pwrite64(int, const void *, size_t, off64_t);
596 extern off64_t tell64(int);
597 extern int truncate64(const char *, off64_t);
598 extern int lockf64(int, int, off64_t);
599 #endif /* __LARGEFILE64_SOURCE */
600 #if !defined(_XOPEN_OR_POSIX) || defined(_XPG4_2) || defined(__EXTENSIONS__)
601 #pragma unknown_control_flow(vfork)
602 #endif /* !defined(_XOPEN_OR_POSIX) || defined(_XPG4_2)... */
603 /*
604  * getlogin_r() & ttynname_r() prototypes are defined here.
605  */
606 /*
607  * Previous releases of Solaris, starting at 2.3, provided definitions of
608  * various functions as specified in POSIX.1c, Draft 6. For some of these
609  * functions, the final POSIX 1003.1c standard had a different number of
610  * arguments and return values.
611 /*
612  * The following segment of this header provides support for the standard
613  * interfaces while supporting applications written under earlier
614  * releases. The application defines appropriate values of the feature
615  * test macros __POSIX_C_SOURCE and __POSIX_PTHREAD_SEMANTICS to indicate
616  * whether it was written to expect the Draft 6 or standard versions of
617  * these interfaces, before including this header. This header then
618  * provides a mapping from the source version of the interface to an
619  * appropriate binary interface. Such mappings permit an application
620  * to be built from libraries and objects which have mixed expectations
621  * of the definitions of these functions.
622 /*
623  * For applications using the Draft 6 definitions, the binary symbol is the
624  * same as the source symbol, and no explicit mapping is needed. For the
625  * standard interface, the function func() is mapped to the binary symbol
626  * _posix_func(). The preferred mechanism for the remapping is a compiler
627  * #pragma. If the compiler does not provide such a #pragma, the header file
628  * defines a static function func() which calls the _posix_func() version;
629  * this has to be done instead of #define since POSIX specifies that an
630  * application can #undef the symbol and still be bound to the correct
631  * implementation. Unfortunately, the statics confuse lint so we fallback to
632  * #define in that case.
633 /*
634  * NOTE: Support for the Draft 6 definitions is provided for compatibility
635  * only. New applications/libraries should use the standard definitions.
636 */
637 #if defined(__EXTENSIONS__) || defined(__REENTRANT) || \
638     !defined(_XOPEN_OR_POSIX) || (__POSIX_C_SOURCE - 0 >= 199506L) || \
639     defined(__POSIX_PTHREAD_SEMANTICS)
640 #if __POSIX_C_SOURCE - 0 >= 199506L || defined(__POSIX_PTHREAD_SEMANTICS)
641 #if __USE_LEGACY_LOGINNAME_
642 #ifdef __PRAGMA_REDEFINE_EXTNAME
643 #pragma redefine_extname getlogin_r __posix_getloginx_r

```

```
646 extern int getlogin_r(char *, int);
647 #else /* __PRAGMA_REDEFINE_EXTNAME */
648 extern int __posix_getloginx_r(char *, int);
649 #define getlogin_r __posix_getloginx_r
650 #endif /* __PRAGMA_REDEFINE_EXTNAME */
651 #else /* __USE_LEGACY_LOGNAME */
652 #ifdef __PRAGMA_REDEFINE_EXTNAME
653 #pragma redefine_extname getlogin_r __posix_getlogin_r
654 extern int getlogin_r(char *, int);
655 #else /* __PRAGMA_REDEFINE_EXTNAME */
656 extern int __posix_getlogin_r(char *, int);

658 #ifdef __lint
660 #define getlogin_r __posix_getlogin_r
662 #else /* !__lint */
664 static int
665 getlogin_r(char *__name, int __len)
666 {
667     return (__posix_getlogin_r(__name, __len));
668 }
unchanged_portion_omitted
```

```
*****
5907 Sun Mar 29 19:44:28 2015
new/usr/src/man/man2/vfork.2
5776 vfork and getwd should not be exposed under XPG7
*****
1 .\" Copyright 2014 Garrett D'Amore <garrett@damore.org>
2 .\" te
3 .\" Copyright (c) 2004, Sun Microsystems, Inc. All Rights Reserved.
4 .\" Copyright (c) 1989 AT&T.
5 .\" All rights reserved. The Berkeley software License Agreement
6 .\" specifies the terms and conditions for redistribution.
7 .Dd Aug 20, 2014
8 .Dt VFORK 2
9 .Os
10 .Sh NAME
11 .Nm vfork ,
12 .Nm vforkx
13 .Nd spawn new process in a virtual memory efficient way
14 .Sh SYNOPSIS
15 .In unistd.h
16 .Ft pid_t
17 .Fn vfork void
18 .
19 .In sys/fork.h
20 .Ft pid_t
21 .Fn vforkx "int flags"
22 .Sh DESCRIPTION
23 The
24 .Fn vfork
25 and
26 .Fn vforkx
27 functions create a new process without
4 ." Copyright (c) 1980 Regents of the University of California. All rights rese
5 .TH VFORK 2 "Dec 13, 2006"
6 .SH NAME
7 vfork, vforkx \- spawn new process in a virtual memory efficient way
8 .SH SYNOPSIS
9 .LP
10 .nf
11 #include <unistd.h>
13 \fBpid_t\fR \fBvfork\fR(\fBvoid\fR);
14 .fi

16 .LP
17 .nf
18 #include <sys/fork.h>

20 \fBpid_t\fR \fBvforkx\fR(\fBint\fR \fIflags\fR);
21 .fi

23 .SH DESCRIPTION
24 .sp
25 .LP
26 The \fBvfork()\fR and \fBvforkx()\fR functions create a new process without
28 fully copying the address space of the old process. These functions are useful
29 in instances where the purpose of a
30 .Xr fork 2
31 operation is to create a new
32 system context for an
33 .Xr exec 2
34 operation.
35 .Lp
36 Unlike with the
37 .Fn fork
```

```
38 function, the child process borrows the parent's
39 memory and thread of control until a call to
40 .Fn execve
41 or an exit
42 .Pq either abnormally or by a call to .Xr _exit 2 .
43 Any modification
28 in instances where the purpose of a \fBfork\fR(2) operation is to create a new
29 system context for an \fBexecve()\fR operation (see \fBexec\fR(2)).
30 .sp
31 .LP
32 Unlike with the \fBfork()\fR function, the child process borrows the parent's
33 memory and thread of control until a call to \fBexecve()\fR or an exit (either
34 abnormally or by a call to \fB_exit()\fR (see \fBexit\fR(2)). Any modification
44 made during this time to any part of memory in the child process is reflected
45 in the parent process on return from
46 .Fn vfork
47 or
48 .Fn vforkx .
49 The parent process is suspended while the child is using its resources.
50 .Lp
51 In a multithreaded application,
52 .Fn vfork
53 and
54 .Fn vforkx
55 borrow only the thread of control that called
56 .Fn vfork
57 or
58 .Fn vforkx
59 in the parent; that is, the child contains only one thread. The use of
60 .Fn vfork
61 or
62 .Fn vforkx
63 in multithreaded applications, however, is unsafe due to race
36 in the parent process on return from \fBvfork()\fR or \fBvforkx()\fR. The
37 parent process is suspended while the child is using its resources.
38 .sp
39 .LP
40 In a multithreaded application, \fBvfork()\fR and \fBvforkx()\fR borrow only
41 the thread of control that called \fBvfork()\fR or \fBvforkx()\fR in the
42 parent; that is, the child contains only one thread. The use of \fBvfork()\fR
43 or \fBvforkx()\fR in multithreaded applications, however, is unsafe due to race
64 conditions that can cause the child process to become deadlocked and
65 consequently block both the child and parent process from execution
66 indefinitely.
67 .Lp
68 The
69 .Fn vfork
70 and
71 .Fn vforkx
72 functions can normally be used the same way as
73 .Fn fork
74 and
75 .Fn forkx ,
76 respectively. The calling procedure,
47 .sp
48 .LP
49 The \fBvfork()\fR and \fBvforkx()\fR functions can normally be used the same
50 way as \fBfork()\fR and \fBforkx()\fR, respectively. The calling procedure,
77 however, should not return while running in the child's context, since the
78 eventual return from
79 .Fn vfork
80 or
81 .Fn vforkx
82 in the parent would be to
83 a stack frame that no longer exists. The
84 .Fn _exit
```

```

85 function should be used
86 in favor of
87 .Xr exit 3C
88 if unable to perform an
89 .Fn execve
90 operation, since
91 .Fn exit
92 will invoke all functions registered by
93 .Xr atexit 3C
94 and will flush and close standard I/O channels, thereby corrupting the parent
95 eventual return from \fBvfork()\fR or \fBvforkx()\fR in the parent would be to
96 a stack frame that no longer exists. The \fB_exit()\fR function should be used
97 in favor of \fBexit(3C) if unable to perform an \fBexecve()\fR operation,
98 since \fBexit()\fR will invoke all functions registered by \fBatexit()\fR(3C) and
99 will flush and close standard I/O channels, thereby corrupting the parent
100 process's standard I/O data structures. Care must be taken in the child process
101 not to modify any global or local data that affects the behavior of the parent
102 process on return from
103 .Fn vfork
104 or
105 .Fn vforkx ,
106 unless such an effect
107 process on return from \fBvfork()\fR or \fBvforkx()\fR, unless such an effect
108 is intentional.
109 .Lp
110 Unlike
111 .Fn fork
112 and
113 .Fn forkx ,
114 fork handlers are not run when
115 .Fn vfork
116 and
117 .Fn vforkx
118 functions are deprecated. Their sole
119 .sp
120 .LP
121 Unlike \fBfork()\fR and \fBforkx()\fR, fork handlers are not run when
122 \fBvfork()\fR and \fBvforkx()\fR are called.
123 .sp
124 .LP
125 The \fBvfork()\fR and \fBvforkx()\fR functions are deprecated. Their sole
126 legitimate use as a prelude to an immediate call to a function from the
127 .Xr exec 2
128 family can be achieved safely by
129 .Xr posix_spawn 3C
130 or
131 .Xr posix_spawnp 3C .
132 .Ss "Fork Extensions"
133 The
134 .Fn vforkx
135 function accepts a
136 .Fa flags
137 argument consisting of a
138 \fBexec\fR family can be achieved safely by \fBposix_spawn\fR(3C) or
139 \fBposix_spawnp\fR(3C).
140 .SS "Fork Extensions"
141 .sp
142 .LP
143 The \fBvforkx()\fR function accepts a \fIflags\fR argument consisting of a
144 bitwise inclusive-OR of zero or more of the following flags, which are defined

```

```

132 in the header
133 .In sys/fork.h :
134 .Lp
135 .Bl -item -compact -offset indent
136 .It
137 .Dv FORK_NOSIGCHLD
138 .It
139 .Dv FORK_WAITPID
140 .El
141 .Lp
142 See
143 .Xr fork 2
144 for descriptions of these flags. If the
145 .Fa flags
146 argument is 0,
147 .Fn vforkx
148 is identical to
149 .Fn vfork .
150 .Sh RETURN VALUES
151 Upon successful completion,
152 .Fn vfork
153 and
154 .Fn vforkx
155 return 0 to
156 the child process and return the process ID of the child process to the parent
157 process. Otherwise, \fB(mil\fR is returned to the parent process, no child
158 process is created, and
159 .Va errno
160 is set to indicate the error.
161 .Sh ERRORS
162 The
163 .Fn vfork
164 and
165 .Fn vforkx
166 functions will fail if:
167 .Bl -tag -width Er
168 .It Er EAGAIN
169 76 in the header \fB<sys/fork.h>\fR:
170 .br
171 .in +2
172 \fBFORK_NOSIGCHLD\fR
173 .in -2
174 .br
175 .in +2
176 \fBFORK_WAITPID\fR
177 .in -2
178 .sp
179 .LP
180 See \fBfork\fR(2) for descriptions of these flags. If the \fIflags\fR argument
181 is 0, \fBvforkx()\fR is identical to \fBvfork()\fR.
182 .Sh RETURN VALUES
183 .sp
184 .LP
185 Upon successful completion, \fBvfork()\fR and \fBvforkx()\fR return \fB0\fR to
186 the child process and returns the process ID of the child process to the parent
187 process. Otherwise, \fB(mil\fR is returned to the parent process, no child
188 process is created, and \fBerrno\fR is set to indicate the error.
189 .Sh ERRORS
190 .sp
191 .LP
192 The \fBvfork()\fR and \fBvforkx()\fR functions will fail if:
193 .ne 2
194 .na
195 .ad

```

```

105 .RS 10n
169 The system-imposed limit on the total number of processes under execution
170 (either system-quality or by a single user) would be exceeded. This limit is
171 determined when the system is generated.
172 .
173 .It Er ENOMEM
109 .RE

111 .sp
112 .ne 2
113 .na
114 \fB\fBENOMEM\fR\fR
115 .ad
116 .RS 10n
174 There is insufficient swap space for the new process.
175 .El
176 .Lp
177 The
178 .Fn vforkx
179 function will fail if:
180 .Bl -tag -width Er
181 .It Er EINVAL
182 The
183 .va flags
184 argument is invalid.
185 .El
186 .Sh INTERFACE STABILITY
187 The
188 .Fn vfork
189 function is
190 .Sy Obsolete Standard .
191 .Lp
192 The
193 .Fn vforkx
194 function is
195 .Sy Obsolete Uncommitted .
196 .Sh MT-LEVEL
197 .Sy Unsafe .
198 .Sh SEE ALSO
199 .Xr exec 2 ,
200 .Xr exit 2 ,
201 .Xr fork 2 ,
202 .Xr ioctl 2 ,
203 .Xr atexit 3C ,
204 .Xr exit 3C ,
205 .Xr posix_spawn 3C ,
206 .Xr posix_spawnp 3C ,
207 .Xr signal.h 3HEAD ,
208 .Xr wait 3C ,
209 .Xr standards 5
210 .Sh NOTES
211 .RE

212 .sp
213 .LP
214 The \fBvforkx()\fR function will fail if:
215 .sp
216 .ne 2
217 .na
218 \fB\fBEINVAL\fR\fR
219 .ad
220 .RS 10n
221 The \fIflags\fR argument is invalid.
222 .RE

223 .SH ATTRIBUTES

```

```

133 .sp
134 .LP
135 See \fBattributes\fR(5) for descriptions of the following attributes:
136 .sp

138 .sp
139 .TS
140 box;
141 c / c
142 l / l .
143 ATTRIBUTE TYPE ATTRIBUTE VALUE
144 -
145 Interface Stability Obsolete
146 -
147 MT-Level Unsafe
148 .TE

150 .SH SEE ALSO
151 .sp
152 .LP
153 \fBexec\fR(2), \fBexit\fR(2), \fBfork\fR(2), \fBioctl\fR(2), \fBatexit\fR(3C),
154 \fBexit\fR(3C), \fBposix_spawn\fR(3C), \fBposix_spawnp\fR(3C),
155 \fBsignal.h\fR(3HEAD), \fBwait\fR(3C), \fBattributes\fR(5), \fBstandards\fR(5)
156 .Sh NOTES
157 .sp
158 .LP
211 To avoid a possible deadlock situation, processes that are children in the
212 middle of a
213 .Fn vfork
214 or
215 .Fn vforkx
216 are never sent
217 .Dv SIGTTOUT
218 or
219 .Dv SIGTTIN
220 signals; rather, output or ioctls are allowed and input attempts
221 result in an
222 .Dv EOF
223 indication.
224 .Lp
225 To forestall parent memory corruption due to race conditions with signal
226 handling,
227 .Fn vfork
228 and
229 .Fn vforkx
230 treat signal handlers in the child
231 process in the same manner as the
232 .Xr exec 2
233 functions: signals set to be
234 caught by the parent process are set to the default action
235 .Pq Dv SIG_DFL
236 in the child process
237 .Pq see Xr signal.h 3HEAD .
238 Any attempt to set a signal
239 handler in the child before
240 .Fn execve
241 to anything other than
242 .Dv SIG_DFL
243 or
244 .Dv SIG_IGN
245 is disallowed and results in setting the handler to

```

```
246 .Dv SIG_DFL .
247 .Lp
248 On some systems, the implementation of
249 .Fn vfork
250 and
251 .Fn vforkx
252 cause
166 handling, \fBvfork()\fR and \fBvforkx()\fR treat signal handlers in the child
167 process in the same manner as the \fBexec\fR(2) functions: signals set to be
168 caught by the parent process are set to the default action (\fBSIG_DFL\fR) in
169 the child process (see \fBsignal.h\fR(3HEAD)). Any attempt to set a signal
170 handler in the child before \fBexecve()\fR to anything other than \fBSIG_DFL\fR
171 or \fBSIG_IGN\fR is disallowed and results in setting the handler to
172 \fBSIG_DFL\fR.
173 .sp
174 .LP
175 On some systems, the implementation of \fBvfork()\fR and \fBvforkx()\fR cause
253 the parent to inherit register values from the child. This can create problems
254 for certain optimizing compilers if
255 .In unistd.h
256 is not included in the source calling
257 .Fn vfork
258 or if
259 .In sys/fork.h
260 is not included in the
261 source calling
262 .Fn vforkx .
263 .Sh STANDARDS
264 The
265 .Fn vfork
266 function is available in the following compilation environments. See
267 .Xr standards 5 .
268 .Lp
269 .Bl -bullet -compact
270 .It
271 .St -xpg4.2
272 .It
273 .St -susv2
274 .It
275 .St -susv3
276 .El
277 .Lp
278 It was marked obsolete in
279 .St -susv3
280 and removed from
281 .St -p1003.1-2008 .
282 .Lp
283 The
284 .Fn vforkx
285 function is a local extension and not available in any strictly
286 standards-compliant compilation environment.
177 for certain optimizing compilers if \fB<unistd.h>\fR is not included in the
178 source calling \fBvfork()\fR or if \fB<sys/fork.h>\fR is not included in the
179 source calling \fBvforkx()\fR.
```

```
*****
3465 Sun Mar 29 19:44:32 2015
new/usr/src/man/man3c/getwd.3c
5776 vfork and getwd should not be exposed under XPG7
*****
1 .\" Copyright 2015 Garrett D'Amore <garrett@damore.org>
1 '\"
2 .\" Copyright (c) 1992, X/Open Company Limited All Rights Reserved Portions Cop
3 .\" Sun Microsystems, Inc. gratefully acknowledges The Open Group for permission
4 .\" http://www.opengroup.org/bookstore/.
5 .\" The Institute of Electrical and Electronics Engineers and The Open Group, ha
6 .\" This notice shall appear on any product containing this material.
7 .\" The contents of this file are subject to the terms of the Common Development
8 .\" You can obtain a copy of the license at usr/src/OPENSOLARIS.LICENSE or http:
9 .\" When distributing Covered Code, include this CDDL HEADER in each file and in
10 .Dd Mar 30, 2015
11 .Dt GETWD 3C
12 .Os
13 .Sh NAME
14 .Nm getwd
15 .Nd get current working directory pathname
16 .Sh SYNOPSIS
17 .In unistd.h
18 .Ft "char *"
19 .Fn getwd "char *path_name"
20 .Sh DESCRIPTION
21 The
22 .Fn getwd
23 function determines an absolute pathname of the current
24 .Th GETWD 3C "Jul 24, 2002"
25 .Sh NAME
26 getwd \- get current working directory pathname
27 .Sh SYNOPSIS
28 .Lp
29 .nf
30 #include <unistd.h>
31
32 \fBchar *\fR\fBgetwd\fR(\fBchar *\fR\fIpath_name\fR);
33 .fi
34
35 .Sh DESCRIPTION
36 .Sp
37 .Lp
38 The \fBgetwd()\fR function determines an absolute pathname of the current
39 working directory of the calling process, and copies that pathname into the
40 array pointed to by the
41 .Fa path_name
42 argument.
43 .Lp
44 array pointed to by the \fIpath_name\fR argument.
45 .Sp
46 .Lp
47 If the length of the pathname of the current working directory is greater than
48 .Pq Dv PATH_MAX + 1
49 including the null byte,
50 .Fn getwd
51 fails and returns a null pointer.
52 .Sh RETURN VALUES
53 (\fIpath_name\fR + 1) including the null byte, \fBgetwd()\fR fails and returns a
54 null pointer.
55 .Sh RETURN VALUES
56 .Sp
57 .Lp
58 Upon successful completion, a pointer to the string containing the absolute
59 pathname of the current working directory is returned. Otherwise,
60 .Fn getwd
```

```
38 returns a null pointer and the contents of the array pointed to by
39 .Fa path_name
40 are undefined.
41 .Sh ERRORS
42 \fBgetwd()\fR returns a null pointer and the contents of the array pointed to
43 by \fIpath_name\fR are undefined.
44 .Sh ERRORS
45 .Sp
46 .Lp
47 No errors are defined.
48 .Sh USAGE
49 The
50 .Fn getwd
51 function is supplied for backwards compatibility. The
52 .Xr getcwd 3C
53 should be used instead.
54 .Sh INTERFACE STABILITY
55 .Sy Obsolete Standard .
56 .Sh SEE ALSO
57 .Xr getcwd 3C ,
58 .Xr standards 5
59 .Sh STANDARDS
60 The
61 .Fn getwd
62 function is available in the following compilation environments. See
63 .Xr standards 5 .
64 .Lp
65 .Bl -bullet -compact
66 .It
67 .St -xpg4.2
68 .It
69 .St -susv2
70 .It
71 .St -susv3
72 .It
73 .St -susv4
74 .El
75 .Lp
76 It was marked obsolete in
77 .St -susv3
78 and removed from
79 .St -p1003.1-2008 .
80 .Sh USAGE
81 .Sp
82 .Lp
83 For portability to implementations conforming to versions of the X/Open
84 Portability Guide prior to SUS, \fBgetcwd\fR(3C) is preferred over this
85 function.
86 .Sh ATTRIBUTES
87 .Sp
88 .Lp
89 See \fBattributes\fR(5) for descriptions of the following attributes:
90 .Sp
91 .Lp
92 .Ts box;
93 .C
94 .Lp
95 .Lc
96 .Lc
97 .Lc
98 .Lc
99 .Lc
100 .Lc
101 .Lc
102 .Lc
103 .Lc
104 .Lc
105 .Lc
106 .Lc
107 .Lc
108 .Lc
109 .Lc
110 .Lc
111 .Lc
112 .Lc
113 .Lc
114 .Lc
115 .Lc
116 .Lc
117 .Lc
118 .Lc
119 .Lc
120 .Lc
121 .Lc
122 .Lc
123 .Lc
124 .Lc
125 .Lc
126 .Lc
127 .Lc
128 .Lc
129 .Lc
130 .Lc
131 .Lc
132 .Lc
133 .Lc
134 .Lc
135 .Lc
136 .Lc
137 .Lc
138 .Lc
139 .Lc
140 .Lc
141 .Lc
142 .Lc
143 .Lc
144 .Lc
145 .Lc
146 .Lc
147 .Lc
148 .Lc
149 .Lc
150 .Lc
151 .Lc
152 .Lc
153 .Lc
154 .Lc
155 .Lc
156 .Lc
157 .Lc
158 .Lc
159 .Lc
160 .Lc
161 .Lc
162 .Lc
163 .Lc
164 .Lc
165 .Lc
166 .Lc
167 .Lc
168 .Lc
169 .Lc
170 .Lc
171 .Lc
172 .Lc
173 .Lc
174 .Lc
175 .Lc
176 .Lc
177 .Lc
178 .Lc
179 .Lc
180 .Lc
181 .Lc
182 .Lc
183 .Lc
184 .Lc
185 .Lc
186 .Lc
187 .Lc
188 .Lc
189 .Lc
190 .Lc
191 .Lc
192 .Lc
193 .Lc
194 .Lc
195 .Lc
196 .Lc
197 .Lc
198 .Lc
199 .Lc
200 .Lc
201 .Lc
202 .Lc
203 .Lc
204 .Lc
205 .Lc
206 .Lc
207 .Lc
208 .Lc
209 .Lc
210 .Lc
211 .Lc
212 .Lc
213 .Lc
214 .Lc
215 .Lc
216 .Lc
217 .Lc
218 .Lc
219 .Lc
220 .Lc
221 .Lc
222 .Lc
223 .Lc
224 .Lc
225 .Lc
226 .Lc
227 .Lc
228 .Lc
229 .Lc
230 .Lc
231 .Lc
232 .Lc
233 .Lc
234 .Lc
235 .Lc
236 .Lc
237 .Lc
238 .Lc
239 .Lc
240 .Lc
241 .Lc
242 .Lc
243 .Lc
244 .Lc
245 .Lc
246 .Lc
247 .Lc
248 .Lc
249 .Lc
250 .Lc
251 .Lc
252 .Lc
253 .Lc
254 .Lc
255 .Lc
256 .Lc
257 .Lc
258 .Lc
259 .Lc
260 .Lc
261 .Lc
262 .Lc
263 .Lc
264 .Lc
265 .Lc
266 .Lc
267 .Lc
268 .Lc
269 .Lc
270 .Lc
271 .Lc
272 .Lc
273 .Lc
274 .Lc
275 .Lc
276 .Lc
277 .Lc
278 .Lc
279 .Lc
280 .Lc
281 .Lc
282 .Lc
283 .Lc
284 .Lc
285 .Lc
286 .Lc
287 .Lc
288 .Lc
289 .Lc
290 .Lc
291 .Lc
292 .Lc
293 .Lc
294 .Lc
295 .Lc
296 .Lc
297 .Lc
298 .Lc
299 .Lc
300 .Lc
301 .Lc
302 .Lc
303 .Lc
304 .Lc
305 .Lc
306 .Lc
307 .Lc
308 .Lc
309 .Lc
310 .Lc
311 .Lc
312 .Lc
313 .Lc
314 .Lc
315 .Lc
316 .Lc
317 .Lc
318 .Lc
319 .Lc
320 .Lc
321 .Lc
322 .Lc
323 .Lc
324 .Lc
325 .Lc
326 .Lc
327 .Lc
328 .Lc
329 .Lc
330 .Lc
331 .Lc
332 .Lc
333 .Lc
334 .Lc
335 .Lc
336 .Lc
337 .Lc
338 .Lc
339 .Lc
340 .Lc
341 .Lc
342 .Lc
343 .Lc
344 .Lc
345 .Lc
346 .Lc
347 .Lc
348 .Lc
349 .Lc
350 .Lc
351 .Lc
352 .Lc
353 .Lc
354 .Lc
355 .Lc
356 .Lc
357 .Lc
358 .Lc
359 .Lc
360 .Lc
361 .Lc
362 .Lc
363 .Lc
364 .Lc
365 .Lc
366 .Lc
367 .Lc
368 .Lc
369 .Lc
370 .Lc
371 .Lc
372 .Lc
373 .Lc
374 .Lc
375 .Lc
376 .Lc
377 .Lc
378 .Lc
379 .Lc
380 .Lc
381 .Lc
382 .Lc
383 .Lc
384 .Lc
385 .Lc
386 .Lc
387 .Lc
388 .Lc
389 .Lc
390 .Lc
391 .Lc
392 .Lc
393 .Lc
394 .Lc
395 .Lc
396 .Lc
397 .Lc
398 .Lc
399 .Lc
400 .Lc
401 .Lc
402 .Lc
403 .Lc
404 .Lc
405 .Lc
406 .Lc
407 .Lc
408 .Lc
409 .Lc
410 .Lc
411 .Lc
412 .Lc
413 .Lc
414 .Lc
415 .Lc
416 .Lc
417 .Lc
418 .Lc
419 .Lc
420 .Lc
421 .Lc
422 .Lc
423 .Lc
424 .Lc
425 .Lc
426 .Lc
427 .Lc
428 .Lc
429 .Lc
430 .Lc
431 .Lc
432 .Lc
433 .Lc
434 .Lc
435 .Lc
436 .Lc
437 .Lc
438 .Lc
439 .Lc
440 .Lc
441 .Lc
442 .Lc
443 .Lc
444 .Lc
445 .Lc
446 .Lc
447 .Lc
448 .Lc
449 .Lc
450 .Lc
451 .Lc
452 .Lc
453 .Lc
454 .Lc
455 .Lc
456 .Lc
457 .Lc
458 .Lc
459 .Lc
460 .Lc
461 .Lc
462 .Lc
463 .Lc
464 .Lc
465 .Lc
466 .Lc
467 .Lc
468 .Lc
469 .Lc
470 .Lc
471 .Lc
472 .Lc
473 .Lc
474 .Lc
475 .Lc
476 .Lc
477 .Lc
478 .Lc
479 .Lc
480 .Lc
481 .Lc
482 .Lc
483 .Lc
484 .Lc
485 .Lc
486 .Lc
487 .Lc
488 .Lc
489 .Lc
490 .Lc
491 .Lc
492 .Lc
493 .Lc
494 .Lc
495 .Lc
496 .Lc
497 .Lc
498 .Lc
499 .Lc
500 .Lc
501 .Lc
502 .Lc
503 .Lc
504 .Lc
505 .Lc
506 .Lc
507 .Lc
508 .Lc
509 .Lc
510 .Lc
511 .Lc
512 .Lc
513 .Lc
514 .Lc
515 .Lc
516 .Lc
517 .Lc
518 .Lc
519 .Lc
520 .Lc
521 .Lc
522 .Lc
523 .Lc
524 .Lc
525 .Lc
526 .Lc
527 .Lc
528 .Lc
529 .Lc
530 .Lc
531 .Lc
532 .Lc
533 .Lc
534 .Lc
535 .Lc
536 .Lc
537 .Lc
538 .Lc
539 .Lc
540 .Lc
541 .Lc
542 .Lc
543 .Lc
544 .Lc
545 .Lc
546 .Lc
547 .Lc
548 .Lc
549 .Lc
550 .Lc
551 .Lc
552 .Lc
553 .Lc
554 .Lc
555 .Lc
556 .Lc
557 .Lc
558 .Lc
559 .Lc
560 .Lc
561 .Lc
562 .Lc
563 .Lc
564 .Lc
565 .Lc
566 .Lc
567 .Lc
568 .Lc
569 .Lc
570 .Lc
571 .Lc
572 .Lc
573 .Lc
574 .Lc
575 .Lc
576 .Lc
577 .Lc
578 .Lc
579 .Lc
580 .Lc
581 .Lc
582 .Lc
583 .Lc
584 .Lc
585 .Lc
586 .Lc
587 .Lc
588 .Lc
589 .Lc
590 .Lc
591 .Lc
592 .Lc
593 .Lc
594 .Lc
595 .Lc
596 .Lc
597 .Lc
598 .Lc
599 .Lc
600 .Lc
601 .Lc
602 .Lc
603 .Lc
604 .Lc
605 .Lc
606 .Lc
607 .Lc
608 .Lc
609 .Lc
610 .Lc
611 .Lc
612 .Lc
613 .Lc
614 .Lc
615 .Lc
616 .Lc
617 .Lc
618 .Lc
619 .Lc
620 .Lc
621 .Lc
622 .Lc
623 .Lc
624 .Lc
625 .Lc
626 .Lc
627 .Lc
628 .Lc
629 .Lc
630 .Lc
631 .Lc
632 .Lc
633 .Lc
634 .Lc
635 .Lc
636 .Lc
637 .Lc
638 .Lc
639 .Lc
640 .Lc
641 .Lc
642 .Lc
643 .Lc
644 .Lc
645 .Lc
646 .Lc
647 .Lc
648 .Lc
649 .Lc
650 .Lc
651 .Lc
652 .Lc
653 .Lc
654 .Lc
655 .Lc
656 .Lc
657 .Lc
658 .Lc
659 .Lc
660 .Lc
661 .Lc
662 .Lc
663 .Lc
664 .Lc
665 .Lc
666 .Lc
667 .Lc
668 .Lc
669 .Lc
670 .Lc
671 .Lc
672 .Lc
673 .Lc
674 .Lc
675 .Lc
676 .Lc
677 .Lc
678 .Lc
679 .Lc
680 .Lc
681 .Lc
682 .Lc
683 .Lc
684 .Lc
685 .Lc
686 .Lc
687 .Lc
688 .Lc
689 .Lc
690 .Lc
691 .Lc
692 .Lc
693 .Lc
694 .Lc
695 .Lc
696 .Lc
697 .Lc
698 .Lc
699 .Lc
700 .Lc
701 .Lc
702 .Lc
703 .Lc
704 .Lc
705 .Lc
706 .Lc
707 .Lc
708 .Lc
709 .Lc
710 .Lc
711 .Lc
712 .Lc
713 .Lc
714 .Lc
715 .Lc
716 .Lc
717 .Lc
718 .Lc
719 .Lc
720 .Lc
721 .Lc
722 .Lc
723 .Lc
724 .Lc
725 .Lc
726 .Lc
727 .Lc
728 .Lc
729 .Lc
730 .Lc
731 .Lc
732 .Lc
733 .Lc
734 .Lc
735 .Lc
736 .Lc
737 .Lc
738 .Lc
739 .Lc
740 .Lc
741 .Lc
742 .Lc
743 .Lc
744 .Lc
745 .Lc
746 .Lc
747 .Lc
748 .Lc
749 .Lc
750 .Lc
751 .Lc
752 .Lc
753 .Lc
754 .Lc
755 .Lc
756 .Lc
757 .Lc
758 .Lc
759 .Lc
760 .Lc
761 .Lc
762 .Lc
763 .Lc
764 .Lc
765 .Lc
766 .Lc
767 .Lc
768 .Lc
769 .Lc
770 .Lc
771 .Lc
772 .Lc
773 .Lc
774 .Lc
775 .Lc
776 .Lc
777 .Lc
778 .Lc
779 .Lc
780 .Lc
781 .Lc
782 .Lc
783 .Lc
784 .Lc
785 .Lc
786 .Lc
787 .Lc
788 .Lc
789 .Lc
790 .Lc
791 .Lc
792 .Lc
793 .Lc
794 .Lc
795 .Lc
796 .Lc
797 .Lc
798 .Lc
799 .Lc
800 .Lc
801 .Lc
802 .Lc
803 .Lc
804 .Lc
805 .Lc
806 .Lc
807 .Lc
808 .Lc
809 .Lc
810 .Lc
811 .Lc
812 .Lc
813 .Lc
814 .Lc
815 .Lc
816 .Lc
817 .Lc
818 .Lc
819 .Lc
820 .Lc
821 .Lc
822 .Lc
823 .Lc
824 .Lc
825 .Lc
826 .Lc
827 .Lc
828 .Lc
829 .Lc
830 .Lc
831 .Lc
832 .Lc
833 .Lc
834 .Lc
835 .Lc
836 .Lc
837 .Lc
838 .Lc
839 .Lc
840 .Lc
841 .Lc
842 .Lc
843 .Lc
844 .Lc
845 .Lc
846 .Lc
847 .Lc
848 .Lc
849 .Lc
850 .Lc
851 .Lc
852 .Lc
853 .Lc
854 .Lc
855 .Lc
856 .Lc
857 .Lc
858 .Lc
859 .Lc
860 .Lc
861 .Lc
862 .Lc
863 .Lc
864 .Lc
865 .Lc
866 .Lc
867 .Lc
868 .Lc
869 .Lc
870 .Lc
871 .Lc
872 .Lc
873 .Lc
874 .Lc
875 .Lc
876 .Lc
877 .Lc
878 .Lc
879 .Lc
880 .Lc
881 .Lc
882 .Lc
883 .Lc
884 .Lc
885 .Lc
886 .Lc
887 .Lc
888 .Lc
889 .Lc
890 .Lc
891 .Lc
892 .Lc
893 .Lc
894 .Lc
895 .Lc
896 .Lc
897 .Lc
898 .Lc
899 .Lc
900 .Lc
901 .Lc
902 .Lc
903 .Lc
904 .Lc
905 .Lc
906 .Lc
907 .Lc
908 .Lc
909 .Lc
910 .Lc
911 .Lc
912 .Lc
913 .Lc
914 .Lc
915 .Lc
916 .Lc
917 .Lc
918 .Lc
919 .Lc
920 .Lc
921 .Lc
922 .Lc
923 .Lc
924 .Lc
925 .Lc
926 .Lc
927 .Lc
928 .Lc
929 .Lc
930 .Lc
931 .Lc
932 .Lc
933 .Lc
934 .Lc
935 .Lc
936 .Lc
937 .Lc
938 .Lc
939 .Lc
940 .Lc
941 .Lc
942 .Lc
943 .Lc
944 .Lc
945 .Lc
946 .Lc
947 .Lc
948 .Lc
949 .Lc
950 .Lc
951 .Lc
952 .Lc
953 .Lc
954 .Lc
955 .Lc
956 .Lc
957 .Lc
958 .Lc
959 .Lc
960 .Lc
961 .Lc
962 .Lc
963 .Lc
964 .Lc
965 .Lc
966 .Lc
967 .Lc
968 .Lc
969 .Lc
970 .Lc
971 .Lc
972 .Lc
973 .Lc
974 .Lc
975 .Lc
976 .Lc
977 .Lc
978 .Lc
979 .Lc
980 .Lc
981 .Lc
982 .Lc
983 .Lc
984 .Lc
985 .Lc
986 .Lc
987 .Lc
988 .Lc
989 .Lc
990 .Lc
991 .Lc
992 .Lc
993 .Lc
994 .Lc
995 .Lc
996 .Lc
997 .Lc
998 .Lc
999 .Lc
1000 .Lc
```



```
new/usr/src/test/libc-tests/cfg/symbols/unistd_h.cfg
```

```
*****
3933 Sun Mar 29 19:44:36 2015
new/usr/src/test/libc-tests/cfg/symbols/unistd_h.cfg
5776 vfork and getwd should not be exposed under XPG7
*****
```

1 #
2 # This file and its contents are supplied under the terms of the
3 # Common Development and Distribution License (" CDDL"), version 1.0.
4 # You may only use this file in accordance with the terms of version
5 # 1.0 of the CDDL.
6 #
7 # A full copy of the text of the CDDL should have accompanied this
8 # source. A copy of the CDDL is also available via the Internet at
9 # http://www.illumos.org/license/CDDL.
10 #
12 # Copyright 2015 Garrett D'Amore <garrett@damore.org>
14 #
16 # Definitions found in unistd.h
18 #
20 #
21 # Types.
22 #
23 type | pid_t | unistd.h | POSIX+ SUS+
25 #
26 # Values.
27 #
28 # Note that the standard requires the user declare environ.
29 # value | environ | char ** | unistd.h | POSIX+ SUS+
30 value | _CS_PATH | int | unistd.h | SUS+
32 value | _CS_POSIX_V6_ILP32_OFF32_CFLAGS | int | unistd.h | SUSv3+
33 value | _CS_POSIX_V6_ILP32_OFF32_LDFLAGS | int | unistd.h | SUSv3+
34 value | _CS_POSIX_V6_ILP32_OFF32_LIBS | int | unistd.h | SUSv3+
35 value | _CS_POSIX_V6_ILP32_OFFBIG_CFLAGS | int | unistd.h | SUSv3+
36 value | _CS_POSIX_V6_ILP32_OFFBIG_LDFLAGS | int | unistd.h | SUSv3+
37 value | _CS_POSIX_V6_ILP32_OFFBIG_LIBS | int | unistd.h | SUSv3+
38 value | _CS_POSIX_V6_LP64_OFF64_CFLAGS | int | unistd.h | SUSv3+
39 value | _CS_POSIX_V6_LP64_OFF64_LDFLAGS | int | unistd.h | SUSv3+
40 value | _CS_POSIX_V6_LP64_OFF64_LIBS | int | unistd.h | SUSv3+
41 value | _CS_POSIX_V6_LPBIK_OFFBIG_CFLAGS | int | unistd.h | SUSv3+
42 value | _CS_POSIX_V6_LPBIK_OFFBIG_LDFLAGS | int | unistd.h | SUSv3+
43 value | _CS_POSIX_V6_LPBIK_OFFBIG_LIBS | int | unistd.h | SUSv3+
44 value | _CS_POSIX_V6_WIDTH_RESTRICTED_ENVS | int | unistd.h | SUSv3+
46 #
47 # Functions
48 #
49 func | access
50 int
51 const char *; int
52 unistd.h | POSIX+ SUS+
54 func | chown
55 int
56 const char *; uid_t; gid_t
57 unistd.h | POSIX+ SUS+
59 func | execl
60 int
61 const char *; const char *

```
1
```

```
new/usr/src/test/libc-tests/cfg/symbols/unistd_h.cfg
```

62 unistd.h | POSIX+ SUS+
64 func | execle
65 int
66 const char *; const char *; char *; char *const []
67 unistd.h | POSIX+ SUS+
69 func | execle
70 int
71 const char *; const char *; char *
72 unistd.h | POSIX+ SUS+
74 func | execve
75 int
76 const char *; char *const []
77 unistd.h | POSIX+ SUS+
79 func | execve
80 int
81 const char *; char *const []
82 unistd.h | POSIX+ SUS+
84 func | execvp
85 int
86 const char *; char *const []
87 unistd.h | POSIX+ SUS+
89 func | fchown
90 int
91 int; uid_t; gid_t
92 unistd.h | -POSIX+ SUS+
94 func | getlogin
95 char *
96 void
97 unistd.h | POSIX+ SUS+
99 func | getlogin_r
100 int
101 char *; size_t
102 unistd.h | -POSIX+ -SUS+ +POSIX-1995+ SUSv2+
104 func | getwd
105 char *
106 char *
107 unistd.h | -ALL SUS+ -SUSv4+
109 func | lchown
110 int
111 const char *; uid_t; gid_t
112 unistd.h | -POSIX+ SUS+
114 func | link
115 int
116 const char *; const char *
117 unistd.h | POSIX+ SUS+
119 # XPG3 may have put this here incorrectly (Open Group says no..., but...)
120 # Probably this is actually our error, and we should kill it, but we can
121 # do that when kill off XPG3 support altogether.
122 func | rename
123 int
124 const char *; const char *
125 unistd.h | -POSIX+ +XPG3 -XPG4+
127 func | symlink

```
2
```

```
128     int                                |\\
129     const char *; const char *          |
130     unistd.h | -XPG3+ -POSIX+ SUS+
132 func | ttynname
133     char *
134     int
135     unistd.h | POSIX+ SUS+
137 func | ttynname_r
138     int
139     int; char *; size_t
140     unistd.h | -POSIX+ -SUS+ +POSIX-1995+ SUSv2+
142 func | ualarm
143     int
144     useconds_t; useconds_t
145     unistd.h | -ALL SUS+ -SUSv4+
147 func | unlink
148     int
149     const char *
150     unistd.h | POSIX+ XPG3+
152 func | usleep
153     int
154     useconds_t
155     unistd.h | -ALL SUS+ -SUSv4+
157 func | vfork
158     pid_t
159     void
160     unistd.h | -ALL SUS+ -SUSv4+
```