```
**********************************************************
   26814 Tue Jun 12 19:54:33 2012
new/exception_lists/packaging
ess_list ioctl now provides all scan results properties for wpa/libdlwlan
first integration of wpa_s control interface client code
first integration of wpa_s wpa_ie parsing code
**********************************************************
   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 2011 Nexenta Systems, Inc.  All rights reserved.
  25 #

  27 #
  28 # Exception List for validate_pkg
  29 #

  31 #
  32 # The following entries are built in the /proto area
  33 # but not included in any packages - this is intentional.
  34 #
  35 usr/include/auth_list.h
  36 usr/include/bsm/audit_door_infc.h
  37 usr/include/bsm/audit_private.h
  38 usr/include/bsm/devalloc.h
  39 usr/include/getxby_door.h
  40 usr/include/passwdutil.h
  41 usr/include/priv_utils.h
  42 usr/include/rpcsvc/daemon_utils.h
  43 usr/include/rpcsvc/svc_dg_priv.h
  44 usr/include/security/pam_impl.h
  45 usr/include/sys/clock_impl.h
  46 usr/include/sys/ieeefp.h
  47 usr/include/sys/winlockio.h
  48 usr/include/scsi/plugins/ses/vendor/sun_impl.h
  49 #
  50 # Private/Internal libraries of the Cryptographic Framework.
  51 #
  52 lib/libkcfd.so
  53 lib/llib-lelfsign
  54 lib/llib-lelfsign.ln
  55 lib/llib-lkcfd
  56 lib/llib-lkcfd.ln
  57 usr/include/libelfsign.h
  58 usr/lib/llib-lsoftcrypto
  59 usr/lib/llib-lsoftcrypto.ln
```



```
  60 usr/lib/amd64/llib-lsoftcrypto.ln       i386
  61 usr/lib/sparcv9/llib-lsoftcrypto.ln     sparc

  63 #
  64 # The following files are used by the DHCP service, the
  65 # standalone's DHCP implementation, and the kernel (nfs_dlboot).
  66 # They contain interfaces which are currently private.
  67 #
  68 usr/include/dhcp_svc_confkey.h
  69 usr/include/dhcp_svc_confopt.h
  70 usr/include/dhcp_svc_private.h
  71 usr/include/dhcp_symbol.h
  72 usr/include/sys/sunos_dhcp_class.h
  73 usr/lib/libdhcpsvc.so
  74 usr/lib/llib-ldhcpsvc
  75 usr/lib/llib-ldhcpsvc.ln
  76 #
  77 # Private MAC driver header files
  78 #
  79 usr/include/inet/iptun.h
  80 usr/include/sys/aggr_impl.h
  81 usr/include/sys/aggr.h
  82 usr/include/sys/dld_impl.h
  83 usr/include/sys/dld_ioc.h
  84 usr/include/sys/dls_impl.h
  85 usr/include/sys/dls.h
  86 usr/include/sys/mac_client_impl.h
  87 usr/include/sys/mac_client.h
  88 usr/include/sys/mac_flow_impl.h
  89 usr/include/sys/mac_impl.h
  90 usr/include/sys/mac_soft_ring.h
  91 usr/include/sys/mac_stat.h
  92 #
  93 # Private GLDv3 userland libraries and headers
  94 #
  95 usr/include/libdladm_impl.h
  96 usr/include/libdlaggr.h
  97 usr/include/libdlether.h
  98 usr/include/libdlflow_impl.h
  99 usr/include/libdlflow.h
 100 usr/include/libdliptun.h
 101 usr/include/libdlmgmt.h
 102 usr/include/libdlsim.h
 103 usr/include/libdlstat.h
 104 usr/include/libdlvnic.h
 105 usr/include/libdlwlan_impl.h
 106 usr/include/libdlwlan.h
 107 #
 108 # Virtual Network Interface Card (VNIC)
 109 #
 110 usr/include/sys/vnic.h
 111 usr/include/sys/vnic_impl.h
 112 #
 113 # Private libipadm lint library and header files
 114 #
 115 usr/include/ipadm_ipmgmt.h
 116 usr/include/ipadm_ndpd.h
 117 usr/include/libipadm.h
 118 lib/llib-lipadm
 119 lib/llib-lipadm.ln
 120 lib/libipadm.so
 121 #
 122 # Private libsocket header file
 123 #
 124 usr/include/libsocket_priv.h
 125 #
```

```
126 # IKE and IPsec support library exceptions.  The IKE support
127 # library contains exclusively private interfaces, as does
128 # libipsecutil.  My apologies for the glut of header files here.
129 #
130 usr/include/errfp.h
131 usr/include/ikedoor.h
132 usr/include/ipsec_util.h
133 usr/lib/libike.so
134 usr/lib/amd64/libike.so                 i386
135 usr/lib/sparcv9/libike.so               sparc
136 usr/lib/libipsecutil.so
137 usr/lib/amd64/libipsecutil.so           i386
138 usr/lib/sparcv9/libipsecutil.so         sparc
139 usr/lib/llib-like
140 usr/lib/llib-like.ln
141 usr/lib/amd64/llib-like.ln              i386
142 usr/lib/sparcv9/llib-like.ln            sparc
143 usr/lib/llib-lipsecutil
144 usr/lib/llib-lipsecutil.ln
145 usr/lib/amd64/llib-lipsecutil.ln        i386
146 usr/lib/sparcv9/llib-lipsecutil.ln      sparc
147 #
148 usr/include/inet/ip_impl.h
149 usr/include/inet/ip_ndp.h
150 usr/include/inet/ip2mac_impl.h
151 usr/include/inet/ip2mac.h
152 usr/include/inet/rawip_impl.h
153 usr/include/inet/tcp_impl.h
154 usr/include/inet/udp_impl.h
155 usr/include/libmail.h
156 usr/include/libnwam_priv.h
157 usr/include/protocols/ripngd.h
158 usr/include/s_string.h
159 usr/include/sys/logindmux_impl.h
160 usr/include/sys/vgareg.h
161 #
162 # Some IPsec headers can’t be shipped lest we hit export controls...
163 #
164 usr/include/inet/ipsec_impl.h
165 usr/include/inet/ipsec_info.h
166 usr/include/inet/ipsecah.h
167 usr/include/inet/ipsecesp.h
168 usr/include/inet/keysock.h
169 usr/include/inet/sadb.h
170 usr/include/sys/sha1_consts.h
171 usr/include/sys/sha2_consts.h
172 #
173 #
174 # Filtering out directories not shipped
175 #
176 usr/4lib                                i386
177 #
178 # These files contain definitions shared privately between the kernel
179 # and libc.  There is no reason for them to be part of a package that
180 # a customer should ever see.  They are installed in the proto area by
181 # the uts build because libc and and other components, like truss, are
182 # dependent upon their contents and should not have their own copies.
183 #
184 usr/include/sys/libc_kernel.h
185 usr/include/sys/synch32.h
186 #
187 # These files are installed in the proto area by the build of libproc for
188 # the benefit of the builds of cmd/truss, cmd/gcore and cmd/ptools, which
189 # use libproc as their common process-control library.  These are not
190 # interfaces for customer use, so the files are excluded from packaging.
191 #
```



```
192 lib/llib-lproc
193 lib/llib-lproc.ln
194 lib/amd64/llib-lproc.ln         i386
195 lib/sparcv9/llib-lproc.ln       sparc
196 usr/include/libproc.h
197 #
198 # Private interfaces for libdisasm
199 #
200 usr/include/libdisasm.h
201 usr/lib/llib-ldisasm
202 usr/lib/llib-ldisasm.ln
203 usr/lib/amd64/llib-ldisasm.ln           i386
204 usr/lib/sparcv9/llib-ldisasm.ln         sparc
205 #
206 # Private interfaces for libraidcfg
207 #
208 usr/include/raidcfg_spi.h
209 usr/include/raidcfg.h
210 usr/lib/libraidcfg.so
211 usr/lib/amd64/libraidcfg.so             i386
212 usr/lib/sparcv9/libraidcfg.so           sparc
213 usr/lib/llib-lraidcfg
214 usr/lib/llib-lraidcfg.ln
215 usr/lib/amd64/llib-lraidcfg.ln          i386
216 usr/lib/sparcv9/llib-lraidcfg.ln        sparc
217 #
218 # This file is used for private communication between mdb, drv/kmdb, and
219 # misc/kmdb.  The interfaces described herein are not intended for customer
220 # use, and are thus excluded from packaging.
221 #
222 usr/include/sys/kmdb.h
223 #
224 # These files are installed in the proto area by the build of libdhcpagent
225 # and libdhcputil for the benefit of DHCP-related networking commands such
226 # as dhcpagent, dhcpinfo, ifconfig, and netstat.  These are not interfaces
227 # for customer use, so the files are excluded from packaging.
228 #
229 lib/libdhcpagent.so
230 lib/libdhcputil.so
231 lib/llib-ldhcpagent
232 lib/llib-ldhcpagent.ln
233 lib/llib-ldhcputil
234 lib/llib-ldhcputil.ln
235 usr/include/dhcp_hostconf.h
236 usr/include/dhcp_impl.h
237 usr/include/dhcp_inittab.h
238 usr/include/dhcp_stable.h
239 usr/include/dhcp_symbol_common.h
240 usr/include/dhcpagent_ipc.h
241 usr/include/dhcpagent_util.h
242 usr/include/dhcpmsg.h
243 usr/lib/libdhcpagent.so
244 usr/lib/libdhcputil.so
245 usr/lib/llib-ldhcpagent
246 usr/lib/llib-ldhcpagent.ln
247 usr/lib/llib-ldhcputil
248 usr/lib/llib-ldhcputil.ln
249 #
250 # These files are installed in the proto area by the build of libinstzones
251 # and libpkg
252 #
253 usr/lib/llib-linstzones
254 usr/lib/llib-linstzones.ln
255 usr/lib/llib-lpkg
256 usr/lib/llib-lpkg.ln
257 #
```

```
258 # Don’t ship header files private to libipmp and in.mpathd
259 #
260 usr/include/ipmp_query_impl.h
261 #
262 # These files are installed in the proto area by the build of libinetsvc,
263 # an inetd-specific library shared by inetd, inetadm and inetconv. Only
264 # the shared object is shipped.
265 #
266 usr/include/inetsvc.h
267 usr/lib/libinetsvc.so
268 usr/lib/llib-linetsvc
269 usr/lib/llib-linetsvc.ln
270 #
271 # These files are installed in the proto area by the build of libinetutil,
272 # a general purpose library for the benefit of internet utilities. Only
273 # the shared object is shipped.
274 #
275 lib/libinetutil.so
276 lib/amd64/libinetutil.so                i386
277 lib/sparcv9/libinetutil.so              sparc
278 lib/llib-linetutil
279 lib/llib-linetutil.ln
280 lib/amd64/llib-linetutil.ln             i386
281 lib/sparcv9/llib-linetutil.ln           sparc
282 usr/include/libinetutil.h
283 usr/include/netinet/inetutil.h
284 usr/include/ofmt.h
285 usr/lib/libinetutil.so
286 usr/lib/amd64/libinetutil.so            i386
287 usr/lib/sparcv9/libinetutil.so          sparc
288 usr/lib/llib-linetutil
289 usr/lib/llib-linetutil.ln
290 usr/lib/amd64/llib-linetutil.ln         i386
291 usr/lib/sparcv9/llib-linetutil.ln       sparc
292 #
293 # Miscellaneous kernel interfaces or kernel<->user interfaces that are
294 # consolidation private and we do not want to export at this time.
295 #
296 usr/include/sys/cryptmod.h
297 usr/include/sys/dumpadm.h
298 usr/include/sys/ontrap.h
299 usr/include/sys/sysmsg_impl.h
300 usr/include/sys/vlan.h
301 #
302 # These files are installed in the proto area so lvm can use
303 # them during the build process.
304 #
305 lib/llib-lmeta
306 lib/llib-lmeta.ln
307 usr/include/sdssc.h
308 usr/lib/llib-lmeta
309 usr/lib/llib-lmeta.ln
310 #
311 # non-public pci header
312 #
313 usr/include/sys/pci_impl.h
314 usr/include/sys/pci_tools.h
315 #
316 # Exception list for RCM project, included by librcm and rcm_daemon
317 #
318 usr/include/librcm_event.h
319 usr/include/librcm_impl.h
320 #
321 # MDB deliverables that are not yet public
322 #
323 usr/lib/mdb/proc/mdb_test.so
```



```
324 usr/lib/mdb/proc/sparcv9/mdb_test.so    sparc
325 #
326 # SNCA project exception list
327 #
328 usr/include/inet/kssl/kssl.h
329 usr/include/inet/kssl/ksslimpl.h
330 usr/include/inet/kssl/ksslproto.h
331 usr/include/inet/nca
332 #
333 # these are "removed" from the source product build because the only
334 # packages that currently deliver them are removed.
335 # they really should’t be in here.
336 #
337 etc/sfw
338 #
339 # Entries for the libmech_krb5 symlink, which has been included
340 # for build purposes only, not delivered to customers.
341 #
342 usr/lib/gss/libmech_krb5.so
343 usr/lib/amd64/gss/libmech_krb5.so       i386
344 usr/lib/sparcv9/gss/libmech_krb5.so     sparc
345 usr/lib/libmech_krb5.so
346 usr/lib/amd64/libmech_krb5.so           i386
347 usr/lib/sparcv9/libmech_krb5.so         sparc
348 #
349 # Entries for headers from efcode project which user does not need to see
350 #
351 usr/platform/sun4u/include/sys/fc_plat.h                        sparc
352 usr/platform/sun4u/include/sys/fcode.h                          sparc
353 #
354 # Private net80211 headers
355 #
356 usr/include/sys/net80211_crypto.h
357 usr/include/sys/net80211_ht.h
358 usr/include/sys/net80211_proto.h
359 usr/include/sys/net80211.h
360 #
361 usr/include/net/wpa.h
362 #
361 # PPPoE files not delivered to customers.
362 #
363 usr/include/net/pppoe.h
364 usr/include/net/sppptun.h
365 #
366 # Simnet
367 #
368 usr/include/net/simnet.h
369 #
370 # Bridging internal data structures
371 #
372 usr/include/net/bridge_impl.h
373 #
374 # User<->kernel interface used by cfgadm/USB only
375 #
376 usr/include/sys/usb/hubd/hubd_impl.h
377 #
378 # User<->kernel interface used by cfgadm/SATA only
379 #
380 usr/include/sys/sata/sata_cfgadm.h                      i386
381 #
382 # Private ucred kernel header
383 #
384 usr/include/sys/ucred.h
385 #
386 # Private and/or platform-specific smf(5) files
387 #
```

```
388 lib/librestart.so
389 lib/llib-lrestart
390 lib/llib-lrestart.ln
391 lib/amd64/llib-lrestart.ln                                  i386
392 lib/sparcv9/llib-lrestart.ln                               sparc
393 usr/include/libcontract_priv.h
394 usr/include/librestart_priv.h
395 usr/include/librestart.h
396 usr/lib/librestart.so
397 usr/lib/sparcv9/librestart.so                               sparc
398 lib/svc/manifest/platform/sun4u                             i386
399 lib/svc/manifest/platform/sun4v                             i386
400 var/svc/manifest/platform/sun4u                             i386
401 var/svc/manifest/platform/sun4v                             i386
402 etc/svc/profile/platform_sun4v.xml                          i386
403 etc/svc/profile/platform_SUNW,SPARC-Enterprise.xml          i386
404 etc/svc/profile/platform_SUNW,Sun-Fire-15000.xml            i386
405 etc/svc/profile/platform_SUNW,Sun-Fire-880.xml            i386
406 etc/svc/profile/platform_SUNW,Sun-Fire-V890.xml           i386
407 etc/svc/profile/platform_SUNW,Sun-Fire.xml                i386
408 etc/svc/profile/platform_SUNW,Ultra-Enterprise-10000.xml  i386
409 etc/svc/profile/platform_SUNW,UltraSPARC-IIe-NetraCT-40.xml  i386
410 etc/svc/profile/platform_SUNW,UltraSPARC-IIe-NetraCT-60.xml  i386
411 etc/svc/profile/platform_SUNW,UltraSPARC-IIi-Netract.xml   i386
412 #
413 # Private libuutil files
414 #
415 lib/libuutil.so
416 lib/llib-luutil
417 lib/llib-luutil.ln
418 lib/sparcv9/llib-luutil.ln              sparc
419 usr/include/libuutil_impl.h
420 usr/lib/libuutil.so
421 usr/lib/sparcv9/libuutil.so             sparc
422 #
423 # Private Multidata file.
424 #
425 usr/include/sys/multidata_impl.h
426 #
427 # The following files are used by wanboot.
428 # They contain interfaces which are currently private.
429 #
430 usr/include/sys/wanboot_impl.h
431 usr/include/wanboot
432 usr/include/wanbootutil.h
433 #
434 # Even though all the objects built under usr/src/stand are later glommed
435 # together into a couple of second-stage boot loaders, we dump the static
436 # archives and lint libraries into $(ROOT)/stand for intermediate use
437 # (e.g., for lint, linking the second-stage boot loaders, ...).  Since
438 # these are merely intermediate objects, they do not need to be packaged.
439 #
440 stand                                   sparc
441 #
442 # Private KCF header files
443 #
444 usr/include/sys/crypto/elfsign.h
445 usr/include/sys/crypto/impl.h
446 usr/include/sys/crypto/ops_impl.h
447 usr/include/sys/crypto/sched_impl.h
448 #
449 # The following files are installed in the proto area
450 # by the build of libavl (AVL Tree Interface Library).
451 # libavl contains interfaces which are all private interfaces.
452 #
453 lib/libavl.so
```



```
454 lib/amd64/libavl.so             i386
455 lib/sparcv9/libavl.so           sparc
456 lib/llib-lavl
457 lib/llib-lavl.ln
458 lib/amd64/llib-lavl.ln          i386
459 lib/sparcv9/llib-lavl.ln        sparc
460 usr/lib/libavl.so
461 usr/lib/amd64/libavl.so         i386
462 usr/lib/sparcv9/libavl.so       sparc
463 usr/lib/llib-lavl
464 usr/lib/llib-lavl.ln
465 usr/lib/amd64/llib-lavl.ln      i386
466 usr/lib/sparcv9/llib-lavl.ln    sparc
467 #
468 # The following files are installed in the proto area
469 # by the build of libcmdutils (Command Utilities Library).
470 # libcmdutils contains interfaces which are all private interfaces.
471 #
472 lib/libcmdutils.so
473 lib/amd64/libcmdutils.so                i386
474 lib/sparcv9/libcmdutils.so              sparc
475 lib/llib-lcmdutils
476 lib/llib-lcmdutils.ln
477 lib/amd64/llib-lcmdutils.ln             i386
478 lib/sparcv9/llib-lcmdutils.ln           sparc
479 usr/include/libcmdutils.h
480 usr/lib/libcmdutils.so
481 usr/lib/amd64/libcmdutils.so            i386
482 usr/lib/sparcv9/libcmdutils.so          sparc
483 usr/lib/llib-lcmdutils
484 usr/lib/llib-lcmdutils.ln
485 usr/lib/amd64/llib-lcmdutils.ln         i386
486 usr/lib/sparcv9/llib-lcmdutils.ln       sparc
487 #
488 # Private interfaces in libsec
489 #
490 usr/include/aclutils.h
491 #
492 # USB skeleton driver stays in sync with the rest of USB but doesn’t ship.
493 #
494 kernel/drv/usbskel              i386
495 kernel/drv/amd64/usbskel        i386
496 kernel/drv/sparcv9/usbskel      sparc
497 kernel/drv/usbskel.conf
498 #
499 # Consolidation and Sun private libdevid interfaces
500 # Public libdevid interfaces provided by devid.h
501 #
502 usr/include/sys/libdevid.h
503 #
504 # The following files are installed in the proto area by the build of
505 # libprtdiag.  libprtdiag contains interfaces which are all private.
506 # Only the shared object is shipped.
507 #
508 usr/platform/sun4u/lib/llib-lprtdiag                            sparc
509 usr/platform/sun4u/lib/llib-lprtdiag.ln                         sparc
510 usr/platform/sun4v/lib/llib-lprtdiag.ln                         sparc
511 #
512 # The following files are installed in the proto area by the build of
513 # mdesc driver in sun4v. These header files are used on in the build
514 # and do not need to be shipped to customers.
515 #
516 usr/include/sys/mdesc.h                                 sparc
517 usr/include/sys/mdesc_impl.h                            sparc
518 usr/platform/sun4v/include/sys/mach_descrip.h           sparc
519 #
```

```
520 # The following files are installed in the proto area by the build of
521 # libpcp.  libpcp contains interfaces which are all private.
522 # Only the shared object is shipped.
523 #
524 usr/platform/sun4v/lib/llib-lpcp.ln                             sparc
525 usr/platform/SUNW,Netra-CP3060/lib/llib-lpcp.ln                 sparc
526 usr/platform/SUNW,Netra-CP3260/lib/llib-lpcp.ln                 sparc
527 usr/platform/SUNW,Netra-T5220/lib/llib-lpcp.ln                  sparc
528 usr/platform/SUNW,Netra-T5440/lib/llib-lpcp.ln                  sparc
529 usr/platform/SUNW,SPARC-Enterprise-T5120/lib/llib-lpcp.ln       sparc
530 usr/platform/SUNW,Sun-Blade-T6300/lib/llib-lpcp.ln             sparc
531 usr/platform/SUNW,Sun-Blade-T6320/lib/llib-lpcp.ln             sparc
532 usr/platform/SUNW,Sun-Fire-T200/lib/llib-lpcp.ln               sparc
533 usr/platform/SUNW,T5140/lib/llib-lpcp.ln                       sparc
534 usr/platform/SUNW,USBRDT-5240/lib/llib-lpcp.ln                 sparc
535 #
536 # ZFS internal tools and lint libraries
537 #
538 usr/lib/llib-lzfs_jni
539 usr/lib/llib-lzfs_jni.ln
540 usr/lib/amd64/llib-lzfs_jni.ln          i386
541 usr/lib/sparcv9/llib-lzfs_jni.ln        sparc
542 usr/lib/llib-lzpool
543 usr/lib/llib-lzpool.ln                  i386
544 usr/lib/amd64/llib-lzpool.ln            i386
545 usr/lib/sparcv9/llib-lzpool.ln          sparc
546 #
547 # ZFS JNI headers
548 #
549 usr/include/libzfs_jni_dataset.h
550 usr/include/libzfs_jni_disk.h
551 usr/include/libzfs_jni_diskmgt.h
552 usr/include/libzfs_jni_ipool.h
553 usr/include/libzfs_jni_main.h
554 usr/include/libzfs_jni_pool.h
555 usr/include/libzfs_jni_property.h
556 usr/include/libzfs_jni_util.h
557 #
558 # These files are installed in the proto area for Solaris scsi_vhci driver
559 # (for MPAPI support) and should not be shipped
560 #
561 usr/include/sys/scsi/adapters/mpapi_impl.h
562 usr/include/sys/scsi/adapters/mpapi_scsi_vhci.h
563 #
564 # This library is installed in the proto area by the build of libdisasm, and is
565 # only used when building the KMDB disasm module.
566 #
567 usr/lib/libstanddisasm.so
568 usr/lib/amd64/libstanddisasm.so         i386
569 usr/lib/sparcv9/libstanddisasm.so       sparc
570 #
571 # TSol: tsol doesn't ship lint source, and tsnet isn't for customers at all.
572 #
573 lib/libtsnet.so
574 usr/lib/llib-ltsnet
575 usr/lib/llib-ltsol
576 #
577 # nss interfaces shared between libnsl and other ON libraries.
578 #
579 usr/include/nss.h
580 #
581 # AT&T AST (ksh93) files which are currently needed only to build OS/Net
582 # (msgcc&co.)
583 # libast
584 usr/lib/libast.so
585 usr/lib/amd64/libast.so                 i386
```



```
586 usr/lib/sparcv9/libast.so               sparc
587 usr/lib/llib-last
588 usr/lib/llib-last.ln
589 usr/lib/amd64/llib-last.ln              i386
590 usr/lib/sparcv9/llib-last.ln            sparc
591 # libcmd
592 usr/lib/llib-lcmd
593 usr/lib/llib-lcmd.ln
594 usr/lib/amd64/llib-lcmd.ln              i386
595 usr/lib/sparcv9/llib-lcmd.ln            sparc
596 # libdll
597 usr/lib/libdll.so
598 usr/lib/amd64/libdll.so                 i386
599 usr/lib/sparcv9/libdll.so               sparc
600 usr/lib/llib-ldll
601 usr/lib/llib-ldll.ln
602 usr/lib/amd64/llib-ldll.ln              i386
603 usr/lib/sparcv9/llib-ldll.ln            sparc
604 # libpp (a helper library needed by AST's msgcc)
605 usr/lib/libpp.so
606 usr/lib/llib-lpp
607 usr/lib/llib-lpp.ln
608 usr/lib/locale/C/LC_MESSAGES/libpp
609 # libshell
610 usr/lib/libshell.so
611 usr/lib/amd64/libshell.so               i386
612 usr/lib/sparcv9/libshell.so             sparc
613 usr/lib/llib-lshell
614 usr/lib/llib-lshell.ln
615 usr/lib/amd64/llib-lshell.ln            i386
616 usr/lib/sparcv9/llib-lshell.ln          sparc
617 # libsum
618 usr/lib/libsum.so
619 usr/lib/amd64/libsum.so                 i386
620 usr/lib/sparcv9/libsum.so               sparc
621 usr/lib/llib-lsum
622 usr/lib/llib-lsum.ln
623 usr/lib/amd64/llib-lsum.ln              i386
624 usr/lib/sparcv9/llib-lsum.ln            sparc
625 #
626 # This file is used in ON to build DSCP clients.  It is not for customers.
627 #
628 usr/include/libdscp.h           sparc
629 #
630 # These files are used by the iSCSI Target and the iSCSI Initiator
631 #
632 usr/include/sys/iscsi_protocol.h
633 usr/include/sys/iscsi_authclient.h
634 usr/include/sys/iscsi_authclientglue.h
635 #
636 # These files are used by the COMSTAR iSCSI target port provider
637 #
638 usr/include/sys/idm
639 usr/include/sys/iscsit/chap.h
640 usr/include/sys/iscsit/iscsi_if.h
641 usr/include/sys/iscsit/isns_protocol.h
642 usr/include/sys/iscsit/radius_packet.h
643 usr/include/sys/iscsit/radius_protocol.h
644 #
645 # libshare is private and the 64-bit sharemgr is not delivered.
646 #
647 usr/lib/libshare.so
648 usr/lib/amd64/libshare.so               i386
649 usr/lib/sparcv9/libshare.so             sparc
650 usr/lib/fs/autofs/libshare_autofs.so
651 usr/lib/fs/autofs/amd64/libshare_autofs.so              i386
```

```
usr/lib/fs/autofs/sparcv9/libshare_autofs.so            sparc
usr/lib/fs/nfs/libshare_nfs.so
usr/lib/fs/nfs/amd64/libshare_nfs.so            i386
usr/lib/fs/nfs/sparcv9/libshare_nfs.so          sparc
usr/lib/fs/smb/libshare_smb.so
usr/lib/fs/smb/amd64/libshare_smb.so            i386
usr/lib/fs/smb/sparcv9/libshare_smb.so          sparc
usr/lib/fs/smbfs/libshare_smbfs.so
usr/lib/fs/smbfs/amd64/libshare_smbfs.so        i386
usr/lib/fs/smbfs/sparcv9/libshare_smbfs.so      sparc
usr/include/libshare_impl.h
usr/include/scfutil.h
#
# These files are installed in the proto area by the build of libpri for
# the benefit of the builds of FMA libldom, Zeus, picld plugins, and/or
# other libpri consumers. However, the libpri interfaces are private to
# Sun (Consolidation Private) and not intended for customer use. So these
# files (the symlink and the lint library) are excluded from packaging.
#
usr/lib/libpri.so               sparc
usr/lib/llib-lpri               sparc
usr/lib/llib-lpri.ln            sparc
usr/lib/sparcv9/libpri.so       sparc
usr/lib/sparcv9/llib-lpri.ln    sparc
#
# These files are installed in the proto area by the build of libds for
# the benefit of the builds of sun4v IO FMA and/or other libds
# consumers.  However, the libds interfaces are private to Sun
# (Consolidation Private) and not intended for customer use. So these
# files (the symlink and the lint library) are excluded from packaging.
#
usr/lib/libds.so                        sparc
usr/lib/sparcv9/libds.so                sparc
usr/lib/llib-lds                        sparc
usr/lib/llib-lds.ln                     sparc
usr/lib/sparcv9/llib-lds.ln             sparc
usr/lib/libdscfg.so
usr/lib/llib-ldscfg.ln
usr/platform/sun4v/include/sys/libds.h  sparc
usr/platform/sun4v/include/sys/vlds.h   sparc
#
# Private/Internal u8_textprep header file. Do not ship.
#
usr/include/sys/u8_textprep_data.h
#
# SQLite is private, used by SMF (svc.configd), idmapd and libsmb.
#
usr/include/sqlite
usr/lib/libsqlite-native.o
usr/lib/libsqlite.o
usr/lib/llib-lsqlite.ln
usr/lib/smbsrv/libsqlite.so
#
# Private/Internal kiconv header files. Do not ship.
#
usr/include/sys/kiconv_big5_utf8.h
usr/include/sys/kiconv_cck_common.h
usr/include/sys/kiconv_cp950hkscs_utf8.h
usr/include/sys/kiconv_emea1.h
usr/include/sys/kiconv_emea2.h
usr/include/sys/kiconv_euckr_utf8.h
usr/include/sys/kiconv_euctw_utf8.h
usr/include/sys/kiconv_gb18030_utf8.h
usr/include/sys/kiconv_gb2312_utf8.h
usr/include/sys/kiconv_hkscs_utf8.h
usr/include/sys/kiconv_ja_jis_to_unicode.h
```



```
usr/include/sys/kiconv_ja_unicode_to_jis.h
usr/include/sys/kiconv_ja.h
usr/include/sys/kiconv_ko.h
usr/include/sys/kiconv_latin1.h
usr/include/sys/kiconv_sc.h
usr/include/sys/kiconv_tc.h
usr/include/sys/kiconv_uhc_utf8.h
usr/include/sys/kiconv_utf8_big5.h
usr/include/sys/kiconv_utf8_cp950hkscs.h
usr/include/sys/kiconv_utf8_euckr.h
usr/include/sys/kiconv_utf8_euctw.h
usr/include/sys/kiconv_utf8_gb18030.h
usr/include/sys/kiconv_utf8_gb2312.h
usr/include/sys/kiconv_utf8_hkscs.h
usr/include/sys/kiconv_utf8_uhc.h
#
# At this time, the ttydefs.cleanup file is only useful on sun4u systems
#
etc/flash/postdeployment/ttydefs.cleanup        i386
#
# This header file is shared only between the power commands and
# ppm/srn modules # and should not be in any package
#
usr/include/sys/srn.h
#
# Private/Internal header files of smbsrv. Do not ship.
#
usr/include/smb
usr/include/smbsrv
#
# Private/Internal dtrace scripts of smbsrv. Do not ship.
#
usr/lib/smbsrv/dtrace
#
# Private/Internal (lint) libraries of smbsrv. Do not ship.
#
usr/lib/reparse/llib-lreparse_smb
usr/lib/reparse/llib-lreparse_smb.ln
usr/lib/smbsrv/llib-lmlrpc
usr/lib/smbsrv/llib-lmlrpc.ln
usr/lib/smbsrv/llib-lmlsvc
usr/lib/smbsrv/llib-lmlsvc.ln
usr/lib/smbsrv/llib-lsmb
usr/lib/smbsrv/llib-lsmb.ln
usr/lib/smbsrv/llib-lsmbns
usr/lib/smbsrv/llib-lsmbns.ln
#
#
# Private/Internal 64-bit libraries of smbsrv. Do not ship.
#
usr/lib/smbsrv/amd64                            i386
usr/lib/smbsrv/sparcv9                          sparc

usr/lib/reparse/amd64/libreparse_smb.so         i386
usr/lib/reparse/amd64/libreparse_smb.so.1       i386
usr/lib/reparse/amd64/llib-lreparse_smb.ln      i386
usr/lib/reparse/sparcv9/libreparse_smb.so       sparc
usr/lib/reparse/sparcv9/libreparse_smb.so.1     sparc
usr/lib/reparse/sparcv9/llib-lreparse_smb.ln    sparc
#
# Private dirent, extended to include flags, for use by SMB server
#
usr/include/sys/extdirent.h
#
# Private header files for vscan service
#
```

```
usr/include/libvscan.h
usr/include/sys/vscan.h
#
# libvscan is private
#
usr/lib/vscan/llib-lvscan
usr/lib/vscan/llib-lvscan.ln
#
# i86hvm is not a full platform.  It is just a home for paravirtualized
# drivers.  There is no usr/ component to this sub-platform, but the
# directory is created in the proto area to keep other tools happy.
#
usr/platform/i86hvm                                             i386
#
# Private sdcard framework headers
#
usr/include/sys/sdcard
#
# libsmbfs is private
#
usr/include/netsmb
usr/lib/libsmbfs.so
usr/lib/amd64/libsmbfs.so               i386
usr/lib/sparcv9/libsmbfs.so             sparc
usr/lib/llib-lsmbfs
usr/lib/llib-lsmbfs.ln
usr/lib/amd64/llib-lsmbfs.ln            i386
usr/lib/sparcv9/llib-lsmbfs.ln          sparc
#
# demo & test program for smbfs (private) ACL support
#
usr/lib/fs/smbfs/chacl
usr/lib/fs/smbfs/lsacl
usr/lib/fs/smbfs/testnp
#
# FC related files
kernel/kmdb/fcip                i386
kernel/kmdb/amd64/fcip          i386
kernel/kmdb/sparcv9/fcip        sparc
kernel/kmdb/fcp                 i386
kernel/kmdb/amd64/fcp           i386
kernel/kmdb/sparcv9/fcp         sparc
kernel/kmdb/fctl                i386
kernel/kmdb/amd64/fctl          i386
kernel/kmdb/sparcv9/fctl        sparc
kernel/kmdb/qlc                 i386
kernel/kmdb/amd64/qlc           i386
kernel/kmdb/sparcv9/qlc         sparc
lib/llib-la5k                   sparc
lib/llib-la5k.ln                sparc
lib/sparcv9/llib-la5k.ln        sparc
lib/llib-lg_fc                  sparc
lib/llib-lg_fc.ln               sparc
lib/sparcv9/llib-lg_fc.ln       sparc
usr/include/a_state.h           sparc
usr/include/a5k.h               sparc
usr/include/exec.h              sparc
usr/include/g_scsi.h            sparc
usr/include/g_state.h           sparc
usr/include/gfc.h               sparc
usr/include/l_common.h          sparc
usr/include/l_error.h           sparc
usr/include/rom.h               sparc
usr/include/stgcom.h            sparc
usr/include/sys/fibre-channel
usr/lib/llib-lHBAAPI
```



```
usr/lib/llib-lHBAAPI.ln
usr/lib/amd64/llib-lHBAAPI.ln   i386
usr/lib/sparcv9/llib-lHBAAPI.ln sparc
#
usr/bin/dscfgcli
usr/bin/sd_diag
usr/bin/sd_stats
usr/include/nsctl.h
usr/include/sys/ncall
usr/include/sys/nsc_ddi.h
usr/include/sys/nsc_thread.h
usr/include/sys/nsctl
usr/include/sys/nskernd.h
usr/include/sys/unistat
usr/lib/libnsctl.so
usr/lib/librdc.so
usr/lib/libunistat.so
usr/lib/llib-lnsctl.ln
usr/lib/llib-lrdc.ln
usr/lib/llib-lunistat.ln
#
# These files are used by the iSCSI initiator only.
# No reason to ship them.
#
usr/include/sys/scsi/adapters/iscsi_door.h
usr/include/sys/scsi/adapters/iscsi_if.h
#
# sbd ioctl hdr
#
usr/include/sys/stmf_sbd_ioctl.h
#
# proxy port provider interface
#
usr/include/sys/pppt_ic_if.h
usr/include/sys/pppt_ioctl.h
#
# proxy daemon lint library
#
usr/lib/llib-lstmfproxy
usr/lib/llib-lstmfproxy.ln
usr/lib/amd64/llib-lstmfproxy.ln        i386
usr/lib/sparcv9/llib-lstmfproxy.ln      sparc
#
# portable object file and dictionary used by libfmd_msg test
#
usr/lib/fm/dict/TEST.dict
usr/lib/locale/C/LC_MESSAGES/TEST.mo
usr/lib/locale/C/LC_MESSAGES/TEST.po
#
# Private idmap RPC protocol
#
usr/include/rpcsvc/idmap_prot.h
usr/include/rpcsvc/idmap_prot.x
#
# Private idmap directory API
#
usr/include/directory.h
#
# librstp is private for bridging
#
usr/include/stp_bpdu.h
usr/include/stp_in.h
usr/include/stp_vectors.h
usr/lib/librstp.so
usr/lib/llib-lrstp
usr/lib/llib-lrstp.ln
```

```
916 #
917 # Private nvfru API
918 #
919 usr/include/nvfru.h
920 #
921 # vrrp
922 #
923 usr/include/libvrrpadm.h
924 usr/lib/libvrrpadm.so
925 usr/lib/amd64/libvrrpadm.so             i386
926 usr/lib/sparcv9/libvrrpadm.so           sparc
927 usr/lib/llib-lvrrpadm
928 usr/lib/llib-lvrrpadm.ln
929 usr/lib/amd64/llib-lvrrpadm.ln          i386
930 usr/lib/sparcv9/llib-lvrrpadm.ln        sparc
931 #
932 # This is only used during the -t tools build
933 #
934 opt/onbld/bin/i386/elfsign      i386
935 opt/onbld/bin/sparc/elfsign     sparc

937 #
938 # Private libdwarf
939 #
940 opt/onbld/lib/i386/libdwarf.so  i386
941 opt/onbld/lib/sparc/libdwarf.so sparc

943 #
944 # Private socket filter API
945 #
946 usr/include/sys/sockfilter.h
947 #
948 # We don't actually validate license action payloads, and the license
949 # staging area is provided as a separate basedir for package
950 # publication.  The net result is that everything therein should be
951 # ignored for packaging validation.
952 #
953 licenses
954 # Libbe is private
955 #
956 usr/include/libbe_priv.h
```

```
**********************************************************
    8613 Tue Jun 12 19:54:34 2012
new/usr/src/Makefile.lint
wpa_supplicant pkg now is created correctly in illumos-gate
wpad renamed to wpa_supplicant
**********************************************************
```

```
#
# 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) 2003, 2010, Oracle and/or its affiliates. All rights reserved.

# include global definitions
include Makefile.master

#
# As pieces are made lint-clean, add them here so the nightly build
# can be used to keep them that way.
#
COMMON_SUBDIRS = \
        cmd/acctadm \
        cmd/asa \
        cmd/amt \
        cmd/audio/audioctl \
        cmd/audio/audiotest \
        cmd/audit \
        cmd/auditconfig \
        cmd/auditd \
        cmd/auditreduce \
        cmd/auditstat \
        cmd/auths \
        cmd/autopush \
        cmd/availdevs \
        cmd/avs \
        cmd/awk \
        cmd/banner \
        cmd/bart \
        cmd/basename \
        cmd/bdiff \
        cmd/bfs \
        cmd/busstat \
        cmd/boot \
        cmd/cal \
        cmd/captoinfo \
        cmd/cat \
        cmd/cdrw \
        cmd/cfgadm \
        cmd/checkeq \
        cmd/checknr \
```



```
        cmd/chgrp \
        cmd/chmod \
        cmd/chown \
        cmd/chroot \
        cmd/clinfo \
        cmd/cmd-crypto \
        cmd/cmd-inet/lib \
        cmd/cmd-inet/lib/netcfgd \
        cmd/cmd-inet/lib/nwamd \
        cmd/cmd-inet/sbin \
        cmd/cmd-inet/usr.bin \
        cmd/cmd-inet/usr.lib/bridged \
        cmd/cmd-inet/usr.lib/dsvclockd \
        cmd/cmd-inet/usr.lib/ilbd \
        cmd/cmd-inet/usr.lib/in.dhcpd \
        cmd/cmd-inet/usr.lib/in.mpathd \
        cmd/cmd-inet/usr.lib/in.ndpd \
        cmd/cmd-inet/usr.lib/inetd \
        cmd/cmd-inet/usr.lib/pppoe \
        cmd/cmd-inet/usr.lib/slpd \
        cmd/cmd-inet/usr.lib/vrrpd \
        cmd/cmd-inet/usr.lib/wpad \
        cmd/cmd-inet/usr.lib/wanboot \
        cmd/cmd-inet/usr.sadm \
        cmd/cmd-inet/usr.sbin \
        cmd/cmd-inet/usr.sbin/ilbadm \
        cmd/cmd-inet/usr.sbin/nwamadm \
        cmd/cmd-inet/usr.sbin/nwamcfg \
        cmd/col \
        cmd/compress \
        cmd/consadm \
        cmd/coreadm \
        cmd/cpc \
        cmd/cpio \
        cmd/crypt \
        cmd/csplit \
        cmd/ctrun \
        cmd/ctstat \
        cmd/ctwatch \
        cmd/date \
        cmd/dd \
        cmd/deroff \
        cmd/devctl \
        cmd/devfsadm \
        cmd/devinfo \
        cmd/devmgmt \
        cmd/devprop \
        cmd/dfs.cmds \
        cmd/diff3 \
        cmd/dis \
        cmd/dirname \
        cmd/diskscan \
        cmd/dispadmin \
        cmd/dladm \
        cmd/dlmgmtd \
        cmd/dtrace \
        cmd/du \
        cmd/dumpadm \
        cmd/dumpcs \
        cmd/echo \
        cmd/eject \
        cmd/emul64ioctl \
        cmd/env \
        cmd/expand \
        cmd/fcinfo \
        cmd/fdetach \
```

```
126         cmd/fdformat \
127         cmd/fdisk \
128         cmd/fgrep \
129         cmd/file \
130         cmd/filebench \
131         cmd/find \
132         cmd/fmthard \
133         cmd/fmtmsg \
134         cmd/fold \
135         cmd/fm \
136         cmd/format \
137         cmd/fs.d/fd \
138         cmd/fs.d/lofs/mount \
139         cmd/fs.d/mntfs \
140         cmd/fs.d/pcfs/mount \
141         cmd/fs.d/proc \
142         cmd/fs.d/tmpfs \
143         cmd/fs.d/udfs/mount \
144         cmd/fs.d/ufs/mount \
145         cmd/fs.d/ufs/fsirand\
146         cmd/fs.d/zfs/fstyp \
147         cmd/fwflash \
148         cmd/fuser \
149         cmd/gcore \
150         cmd/genmsg \
151         cmd/getconf \
152         cmd/getdevpolicy \
153         cmd/getfacl \
154         cmd/getopt \
155         cmd/gettext \
156         cmd/grep \
157         cmd/grep_xpg4 \
158         cmd/groups \
159         cmd/halt \
160         cmd/head \
161         cmd/hostid \
162         cmd/hostname \
163         cmd/hotplug \
164         cmd/hotplugd \
165         cmd/idmap \
166         cmd/init \
167         cmd/intrstat \
168         cmd/ipcrm \
169         cmd/ipcs \
170         cmd/isaexec \
171         cmd/isalist \
172         cmd/iscsiadm \
173         cmd/iscsid \
174         cmd/iscsitsvc \
175         cmd/isns \
176         cmd/itadm \
177         cmd/kbd \
178         cmd/killall \
179         cmd/ldap \
180         cmd/last \
181         cmd/lastcomm \
182         cmd/ldapcachemgr \
183         cmd/line \
184         cmd/link \
185         cmd/locator \
186         cmd/localedef \
187         cmd/lockstat \
188         cmd/lofiadm \
189         cmd/logadm \
190         cmd/logger \
191         cmd/login \
```



```
192         cmd/logins \
193         cmd/ls \
194         cmd/luxadm \
195         cmd/lvm \
196         cmd/machid \
197         cmd/makekey \
198         cmd/mdb \
199         cmd/mesg \
200         cmd/mkdir \
201         cmd/mkfifo \
202         cmd/mkfile \
203         cmd/mkmsgs \
204         cmd/mknod \
205         cmd/mpathadm \
206         cmd/modload \
207         cmd/msgfmt \
208         cmd/msgid \
209         cmd/mt \
210         cmd/mv \
211         cmd/ndmpadm \
212         cmd/ndmpd \
213         cmd/ndmpstat \
214         cmd/newform \
215         cmd/newgrp \
216         cmd/newtask \
217         cmd/nice \
218         cmd/nl \
219         cmd/nohup \
220         cmd/nscd \
221         cmd/od \
222         cmd/pagesize \
223         cmd/passwd \
224         cmd/pathchk \
225         cmd/pbind \
226         cmd/pcidr \
227         cmd/pcitool \
228         cmd/pfexec \
229         cmd/pgrep \
230         cmd/picl/picld \
231         cmd/picl/prtpicl \
232         cmd/plockstat \
233         cmd/pools \
234         cmd/power \
235         cmd/powertop \
236         cmd/printf \
237         cmd/latencytop \
238         cmd/ppgsz \
239         cmd/praudit \
240         cmd/prctl \
241         cmd/priocntl \
242         cmd/profiles \
243         cmd/prstat \
244         cmd/prtconf \
245         cmd/prtdiag \
246         cmd/prtvtoc \
247         cmd/ps \
248         cmd/psradm \
249         cmd/psrinfo \
250         cmd/psrset \
251         cmd/ptools \
252         cmd/pwck \
253         cmd/pwconv \
254         cmd/ramdiskadm \
255         cmd/raidctl \
256         cmd/rcap \
257         cmd/rcm_daemon \
```

```
258         cmd/rctladm \
259         cmd/renice \
260         cmd/rm \
261         cmd/rmdir \
262         cmd/rmformat \
263         cmd/rmt \
264         cmd/roles \
265         cmd/rpcgen \
266         cmd/rpcsvc/rpc.bootparamd \
267         cmd/runat \
268         cmd/savecore \
269         cmd/sbdadm \
270         cmd/sdpadm \
271         cmd/sed \
272         cmd/setpgrp \
273         cmd/smbios \
274         cmd/sgs \
275         cmd/smbsrv \
276         cmd/smserverd \
277         cmd/sort \
278         cmd/split \
279         cmd/srptadm \
280         cmd/srptsvc \
281         cmd/ssh \
282         cmd/stat \
283         cmd/stmfadm \
284         cmd/stmfsvc \
285         cmd/stmsboot \
286         cmd/streams/strcmd \
287         cmd/strings \
288         cmd/su \
289         cmd/sulogin \
290         cmd/svc \
291         cmd/swap \
292         cmd/sync \
293         cmd/syseventadm \
294         cmd/syseventd \
295         cmd/syslogd \
296         cmd/tabs \
297         cmd/tail \
298         cmd/th_tools \
299         cmd/tip \
300         cmd/touch \
301         cmd/tr \
302         cmd/truss \
303         cmd/tty \
304         cmd/tzreload \
305         cmd/uadmin \
306         cmd/ul \
307         cmd/userattr \
308         cmd/users \
309         cmd/utmp_update \
310         cmd/utmpd \
311         cmd/valtools \
312         cmd/vrrpadm \
313         cmd/vt \
314         cmd/wall \
315         cmd/who \
316         cmd/whodo \
317         cmd/wracct \
318         cmd/wusbadm \
319         cmd/xargs \
320         cmd/xstr \
321         cmd/yes \
322         cmd/yppasswd \
323         cmd/zdb \
```



```
324         cmd/zdump \
325         cmd/zfs \
326         cmd/zinject \
327         cmd/zlogin \
328         cmd/zoneadm \
329         cmd/zoneadmd \
330         cmd/zonecfg \
331         cmd/zonename \
332         cmd/zpool \
333         cmd/zlook \
334         cmd/ztest \
335         lib/abi \
336         lib/auditd_plugins \
337         lib/libbe \
338         lib/pylibbe \
339         lib/brand/sn1 \
340         lib/brand/solaris10 \
341         lib/crypt_modules \
342         lib/extendedFILE \
343         lib/libadm \
344         lib/libadutils \
345         lib/libadt_jni \
346         lib/libaio \
347         lib/libavl \
348         lib/libbrand \
349         lib/libbsdmalloc \
350         lib/libbsm \
351         lib/libc \
352         lib/libc_db \
353         lib/libcfgadm \
354         lib/libcmdutils \
355         lib/libcommputil \
356         lib/libcontract \
357         lib/libcryptoutil \
358         lib/libctf \
359         lib/libdevice \
360         lib/libdevid \
361         lib/libdevinfo \
362         lib/libdhcpagent \
363         lib/libdhcpdu \
364         lib/libdhcpsvc \
365         lib/libdhcputil \
366         lib/libdisasm \
367         lib/libdiskmgt \
368         lib/libdladm \
369         lib/libdlpi \
370         lib/libdoor \
371         lib/libdscfg \
372         lib/libdtrace \
373         lib/libefi \
374         lib/libelfsign \
375         lib/libexacct \
376         lib/libfcoe \
377         lib/libgen \
378         lib/libgrubmgmt \
379         lib/libgss \
380         lib/libhotplug \
381         lib/libidmap \
382         lib/libilb \
383         lib/libinetsvc \
384         lib/libinetutil \
385         lib/libinstzones \
386         lib/libipadm \
387         lib/libipmi \
388         lib/libipmp \
389         lib/libipp \
```

```
390         lib/libipsecutil \
391         lib/libiscsit \
392         lib/libkmf \
393         lib/libkstat \
394         lib/liblgrp \
395         lib/liblm \
396         lib/libmalloc \
397         lib/libmapmalloc \
398         lib/libmapid \
399         lib/libmd \
400         lib/libmp \
401         lib/libmtmalloc \
402         lib/libndmp \
403         lib/libnsctl \
404         lib/libnsl \
405         lib/libnvpair \
406         lib/libnwam \
407         lib/libpam \
408         lib/libpctx \
409         lib/libpicl \
410         lib/libpicltree \
411         lib/libpkg \
412         lib/libpool \
413         lib/libproc \
414         lib/libpthread \
415         lib/libraidcfg \
416         lib/librcm \
417         lib/librdc \
418         lib/libreparse \
419         lib/librestart \
420         lib/librstp \
421         lib/librt \
422         lib/libscf \
423         lib/libsec \
424         lib/libsecdb \
425         lib/libsendfile \
426         lib/libsip \
427         lib/libshare \
428         lib/libsldap \
429         lib/libslp \
430         lib/libsmbfs \
431         lib/libsmbios \
432         lib/libsmedia \
433         lib/libsrpt \
434         lib/libstmf \
435         lib/libsun_ima \
436         lib/libsysevent \
437         lib/libthread \
438         lib/libtsnet \
439         lib/libtsol \
440         lib/libumem \
441         lib/libunistat \
442         lib/libuuid \
443         lib/libuutil \
444         lib/libvrrpadm \
445         lib/libwanboot \
446         lib/libwanbootutil \
447         lib/libxnet \
448         lib/libzfs \
449         lib/libzfs_jni \
450         lib/libzonecfg \
451         lib/libzoneinfo \
452         lib/lvm \
453         lib/madv \
454         lib/mpss \
455         lib/nametoaddr \
```



```
456         lib/ncad_addr \
457         lib/nsswitch \
458         lib/pam_modules \
459         lib/passwdutil \
460         lib/pkcs11 \
461         lib/print \
462         lib/raidcfg_plugins \
463         lib/scsi \
464         lib/smbsrv \
465         lib/fm \
466         lib/udapl \
467         lib/watchmalloc \
468         psm \
469         ucbcmd/basename \
470         ucbcmd/biff \
471         ucbcmd/echo \
472         ucbcmd/groups \
473         ucbcmd/mkstr \
474         ucbcmd/printenv \
475         ucbcmd/sum \
476         ucbcmd/test \
477         ucbcmd/users \
478         ucbcmd/whoami

480 i386_SUBDIRS= \
481         cmd/acpihpd \
482         cmd/biosdev \
483         cmd/rtc \
484         cmd/ucodeadm \
485         lib/cfgadm_plugins/sata \
486         lib/cfgadm_plugins/sbd \
487         lib/libfdisk

489 sparc_SUBDIRS= \
490         cmd/datadm \
491         cmd/dcs \
492         cmd/drd \
493         cmd/fruadm \
494         cmd/ldmad \
495         cmd/prtdscp \
496         cmd/prtfru \
497         cmd/sckmd \
498         cmd/virtinfo \
499         cmd/vntsd \
500         lib/libds \
501         lib/libdscp \
502         lib/libpri \
503         lib/libpcp \
504         lib/libtsalarm \
505         lib/libv12n \
506         lib/storage \
507         stand

509 LINTSUBDIRS= $(COMMON_SUBDIRS) $($(MACH)_SUBDIRS)

511 .PARALLEL:      $(LINTSUBDIRS)

513 lint:           uts .WAIT subdirs

515 subdirs:        $(LINTSUBDIRS)

517 uts $(LINTSUBDIRS):     FRC
518         @cd $@; pwd; $(MAKE) lint

520 FRC:
```

```
**********************************************************
    4151 Tue Jun 12 19:54:35 2012
new/usr/src/cmd/Makefile.check
wpa_supplicant pkg now is created correctly in illumos-gate
wpad renamed to wpa_supplicant
**********************************************************
#
# 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, 2010, Oracle and/or its affiliates. All rights reserved.

include ../Makefile.master

#
# Commands providing manifests must offer a check target.  A recursive check
# target across all commands directories is not currently provided.
#
MANIFEST_TOPDIRS=                       \
        acctadm                         \
        auditd                          \
        bnu                             \
        consadm                         \
        coreadm                         \
        cron                            \
        cvcd                            \
        dispadmin                       \
        drd                             \
        dumpadm                         \
        fcinfo                          \
        fcoesvc                         \
        fm                              \
        ibd_upgrade                     \
        intrd                           \
        iscsid                          \
        iscsitsvc                       \
        kbd                             \
        keyserv                         \
        ldapcachemgr                    \
        ldmad                           \
        lms                             \
        dlmgmtd                         \
        ndmpd                           \
        nscd                            \
        oplhpd                          \
        pools                           \
        power                           \
        rexd                            \
        rmvolmgr                        \
```



```
        rpcbind                         \
        rpcsvc                          \
        sa                              \
        saf                             \
        sckmd                           \
        sf880drd                        \
        smserverd                       \
        stmfproxy                       \
        stmfsvc                         \
        stmsboot                        \
        syseventd                       \
        syslogd                         \
        utmpd                           \
        vntsd                           \
        ypcmd                           \
        zoneadm                         \
        zoneadmd

MANIFEST_SUBDIRS=                       \
        boot/scripts                    \
        cmd-crypto/scripts              \
        cmd-inet/usr.lib/ilbd           \
        cmd-inet/usr.lib/in.chargend    \
        cmd-inet/usr.lib/in.daytimed    \
        cmd-inet/usr.lib/in.dhcpd       \
        cmd-inet/usr.lib/in.discardd    \
        cmd-inet/usr.lib/in.echod       \
        cmd-inet/usr.lib/in.ndpd        \
        cmd-inet/usr.lib/in.ripngd      \
        cmd-inet/usr.lib/in.timed       \
        cmd-inet/usr.lib/inetd          \
        cmd-inet/usr.lib/mdnsd          \
        cmd-inet/usr.lib/slpd           \
        cmd-inet/usr.lib/vrrpd          \
        cmd-inet/usr.lib/wpa_supplicant \
        cmd-inet/usr.lib/wpad           \
        cmd-inet/usr.sbin               \
        cmd-inet/usr.sbin/in.ftpd       \
        cmd-inet/usr.sbin/in.rdisc      \
        cmd-inet/usr.sbin/in.routed     \
        cmd-inet/usr.sbin/in.talkd      \
        cmd-inet/usr.sbin/ipsecutils    \
        cmd-inet/usr.sbin/kssl/ksslcfg  \
        cmd-inet/usr.sbin/routeadm      \
        dcs/sparc/sun4u                 \
        dfs.cmds/sharemgr               \
        fs.d/autofs                     \
        fs.d/nfs/svc                    \
        fs.d/smbclnt/svc                \
        gss/gssd                        \
        hal/addons/network-devices      \
        hal/hald/solaris                \
        halt/smf.$(MACH)                \
        hostid/smf                      \
        idmap/idmapd                    \
        ipf/svc                         \
        isns/isnsd                      \
        krb5/kadmin/server              \
        krb5/krb5kdc                    \
        krb5/kwarn                      \
        krb5/slave                      \
        lp/cmd/lpsched                  \
        lvm/rpc.mdcommd                 \
        lvm/rpc.metad                   \
        lvm/rpc.metamedd                \
        lvm/rpc.metamhd                 \
```

```
126         lvm/md_monitord                 \
127         lvm/util                        \
128         picl/picld                      \
129         pools/poold                     \
130         print/bsd-sysv-commands         \
131         print/ppdmgr                    \
132         rcap/rcapd                      \
133         rpcsvc/rpc.bootparamd           \
134         sendmail/lib                    \
135         smbsrv/smbd                     \
136         ssh/etc                         \
137         svc/milestone                   \
138         tsol/labeld                     \
139         tsol/tnctl                      \
140         tsol/tnd                        \
141         tsol/tsol-zones                 \
142         vscan/vscand                    \
143         xvm/ipagent                     \
144         ypcmd/yppasswd                  \
145         ypcmd/ypupdated                 \
146         zonestat/zonestatd

148 $(CLOSED_BUILD)MANIFEST_SUBDIRS +=              \
149         $(CLOSED)/cmd/cmd-inet/usr.lib/in.iked

151 DTEST_SUBDIRS= \
152         dtrace/test/tst

154 .KEEP_STATE:

156 # Manifests cannot be checked in parallel, because we are using the global
157 # repository that is in $(SRC)/cmd/svc/seed/global.db.  This is a
158 # repository that is built from the manifests in this workspace, whereas
159 # the build machine's repository may be out of sync with these manifests.
160 # Because we are using a private repository, svccfg-native must start up a
161 # private copy of configd-native.  We cannot have multiple copies of
162 # configd-native trying to access global.db simultaneously.

164 .NO_PARALLEL:

166 check: svccfg_check $(MANIFEST_TOPDIRS) $(MANIFEST_SUBDIRS) $(DTEST_SUBDIRS)

168 svccfg_check:
169         @$(ECHO) "building requirements for svccfg check ..."; \
170         (cd $(SRC)/cmd/svc/seed && pwd && $(MAKE) $(MFLAGS) global.db)

172 $(MANIFEST_TOPDIRS) $(MANIFEST_SUBDIRS) $(DTEST_SUBDIRS): FRC
173         @cd $@; pwd; $(MAKE) check

175 FRC:
```

****
14850 Tue Jun 12 19:54:36 2012
new/usr/src/cmd/cmd-inet/lib/nwamd/known_wlans.c
secobjs types now are "wep, psk, eap, pin"
dladm_wlan_secmode_t and dladm_secobj_class_t are not related anymore
****
_____***unchanged_portion_omitted_***

```
 248 /*
 249  * Walk security objects looking for one that matches the essid prefix.
 250  * Store the key and keyname if a match is found - we use the last match
 251  * as the key for the known WLAN, since it is the most recently updated.
 252  */
 253 /* ARGSUSED0 */
 254 static boolean_t
 255 find_secobj_matching_prefix(dladm_handle_t dh, void *arg,
 256     const char *secobjname)
 257 {
 258         struct nwamd_secobj_arg *nsa = arg;

 260         if (strncmp(nsa->nsa_essid_prefix, secobjname,
 261             strlen(nsa->nsa_essid_prefix)) == 0) {
 262                 nlog(LOG_DEBUG, "find_secobj_matching_prefix: "
 263                     "found secobj with prefix %s : %s\n",
 264                     nsa->nsa_essid_prefix, secobjname);
 265                 /* Free last key found (if any) */
 266                 if (nsa->nsa_key != NULL)
 267                         free(nsa->nsa_key);
 268                 /* Retrive key so we can get security mode */
 269                 nsa->nsa_key = nwamd_wlan_get_key_named(secobjname, 0);
 270                 (void) strlcpy(nsa->nsa_keyname, secobjname,
 271                     sizeof (nsa->nsa_keyname));
 272                 switch (nsa->nsa_key->wk_class) {
 273                 case DLADM_SECOBJ_CLASS_WEP:
 274                         nsa->nsa_secmode = DLADM_WLAN_SECMODE_WEP;
 275                         nlog(LOG_DEBUG, "find_secobj_matching_prefix: "
 276                             "got WEP key %s", nsa->nsa_keyname);
 277                         break;
 278                 case DLADM_SECOBJ_CLASS_PSK:
 279                         nsa->nsa_secmode = DLADM_WLAN_SECMODE_PSK;
 278                 case DLADM_SECOBJ_CLASS_WPA:
 279                         nsa->nsa_secmode = DLADM_WLAN_SECMODE_WPA;
 280                         nlog(LOG_DEBUG, "find_secobj_matching_prefix: "
 281                             "got WPA key %s", nsa->nsa_keyname);
 282                         break;
 283                 default:
 284                         /* shouldn't happen */
 285                         nsa->nsa_secmode = DLADM_WLAN_SECMODE_NONE;
 286                         nlog(LOG_ERR, "find_secobj_matching_prefix: "
 287                             "key class for key %s was invalid",
 288                             nsa->nsa_keyname);
 289                         break;
 290                 }
 291         }
 292         return (B_TRUE);
 293 }
```

_____***unchanged_portion_omitted_***

```
**********************************************************
   58588 Tue Jun 12 19:54:37 2012
new/usr/src/cmd/cmd-inet/lib/nwamd/ncu_phys.c
secobjs types now are "wep, psk, eap, pin"
dladm_wlan_secmode_t and dladm_secobj_class_t are not related anymore
**********************************************************
______unchanged_portion_omitted_

/* not used */
#endif /* ! codereview */
#define WLAN_ENC(sec)                                           \
        ((sec == DLADM_WLAN_SECMODE_PSK ? "WPA" :               \
        ((sec == DLADM_WLAN_SECMODE_WPA ? "WPA" :               \
        (sec == DLADM_WLAN_SECMODE_WEP ? "WEP" : "none")))

/*
 * NEED_END should return false and when DLADM_WLAN_AUTH_NONE is set
 * (key_mgmt=NONE in wpa_s conf) (!not key_mgmt=WPA-NONE)
 */
#endif /* ! codereview */
#define NEED_ENC(sec)                                           \
        (sec == DLADM_WLAN_SECMODE_PSK || sec == DLADM_WLAN_SECMODE_WEP)
        (sec == DLADM_WLAN_SECMODE_WPA || sec == DLADM_WLAN_SECMODE_WEP)

#define WIRELESS_LAN_INIT_COUNT 8

/*
 * The variable wireless_scan_level specifies the signal level
 * that we will initiate connections to previously-visited APs
 * at when we are in the connected state.
 */
dladm_wlan_strength_t wireless_scan_level = DLADM_WLAN_STRENGTH_WEAK;

/*
 * The variable wireless_scan_interval specifies how often the periodic
 * scan occurs.
 */
uint64_t wireless_scan_interval = WIRELESS_SCAN_INTERVAL_DEFAULT;

/*
 * The variable wireless_autoconf specifies if we use dladm_wlan_autoconf()
 * to connect.
 */
boolean_t wireless_autoconf = B_FALSE;

/*
 * The variable wireless_strict_bssid specifies if we only connect
 * to WLANs with BSSIDs that we previously connected to.
 */
boolean_t wireless_strict_bssid = B_FALSE;

/*
 * We need to ensure scan or connect threads do not run concurrently
 * on any links - otherwise we get radio interference.  Acquire this
 * lock on entering scan/connect threads to prevent this.
 */
pthread_mutex_t wireless_mutex = PTHREAD_MUTEX_INITIALIZER;

static void
scanconnect_entry(void)
{
        (void) pthread_mutex_lock(&wireless_mutex);
}
______unchanged_portion_omitted_

/*
```



```
 * Below are functions used to handle storage/retrieval of keys
 * for a given WLAN. The keys are stored/retrieved using dladm_set_secobj()
 * and dladm_get_secobj().
 */

/*
 * Convert key hexascii string to raw secobj value. This
 * code is very similar to convert_secobj() in dladm.c, it would
 * be good to have a libdladm function to convert values.
 */
static int
key_string_to_secobj_value(char *buf, uint8_t *obj_val, uint_t *obj_lenp,
    dladm_secobj_class_t class)
{
        size_t buf_len = strlen(buf);

        nlog(LOG_DEBUG, "before: key_string_to_secobj_value: buf_len = %d",
            buf_len);
        if (buf_len == 0) {
                /* length zero means "delete" */
                return (0);
        }

        if (buf[buf_len - 1] == '\n')
                buf[--buf_len] = '\0';

        nlog(LOG_DEBUG, "after: key_string_to_secobj_value: buf_len = %d",
            buf_len);

        if (class == DLADM_SECOBJ_CLASS_PSK) {
        if (class == DLADM_SECOBJ_CLASS_WPA) {
                /*
                 * Per IEEE802.11i spec, the Pre-shared key (PSK) length should
                 * be between 8 and 63.
                 */
                if (buf_len < 8 || buf_len > 63) {
                        nlog(LOG_ERR,
                            "key_string_to_secobj_value:"
                            " invalid WPA key length: buf_len = %d", buf_len);
                        return (-1);
                }
                (void) memcpy(obj_val, buf, (uint_t)buf_len);
                *obj_lenp = buf_len;
                return (0);
        }

        switch (buf_len) {
        case 5:         /* ASCII key sizes */
        case 13:
                (void) memcpy(obj_val, buf, (uint_t)buf_len);
                *obj_lenp = (uint_t)buf_len;
                break;
        case 10:
        case 26:        /* Hex key sizes, not preceded by 0x */
                if (hexascii_to_octet(buf, (uint_t)buf_len, obj_val, obj_lenp)
                    != 0) {
                        nlog(LOG_ERR,
                            "key_string_to_secobj_value: invalid WEP key");
                        return (-1);
                }
                break;
        case 12:
        case 28:        /* Hex key sizes, preceded by 0x */
                if (strncmp(buf, "0x", 2) != 0 ||
                    hexascii_to_octet(buf + 2, (uint_t)buf_len - 2, obj_val,
                    obj_lenp) != 0) {
```

```
347                         nlog(LOG_ERR,
348                             "key_string_to_secobj_value: invalid WEP key");
349                         return (-1);
350                 }
351                 break;
352         default:
353                 syslog(LOG_ERR,
354                     "key_string_to_secobj_value: invalid WEP key length");
355                 return (-1);
356         }
357         return (0);
358 }
```

*unchanged_portion_omitted_*

```
406 nwam_error_t
407 nwamd_wlan_set_key(const char *linkname, const char *essid, const char *bssid,
408     uint32_t security_mode, uint_t keyslot, char *raw_key)
409 {
410         nwamd_object_t ncu_obj;
411         nwamd_ncu_t *ncu;
412         nwamd_link_t *link;
413         uint8_t obj_val[DLADM_SECOBJ_VAL_MAX];
414         uint_t obj_len = sizeof (obj_val);
415         char obj_name[DLADM_SECOBJ_NAME_MAX];
416         dladm_status_t status;
417         char errmsg[DLADM_STRSIZE];
418         dladm_secobj_class_t class;

420         if ((ncu_obj = nwamd_ncu_object_find(NWAM_NCU_TYPE_LINK, linkname))
421             == NULL) {
422                 nlog(LOG_ERR, "nwamd_wlan_set_key: could not find object  "
423                     "for link %s", linkname);
424                 return (NWAM_ENTITY_NOT_FOUND);
425         }
426         ncu = ncu_obj->nwamd_object_data;
427         link = &ncu->ncu_link;

429         nlog(LOG_DEBUG, "nwamd_wlan_set_key: running for link %s", linkname);
430         /*
431          * Name key object for this WLAN so it can be later retrieved
432          * (name is unique for each ESSID/BSSID combination).
433          */
434         nwamd_set_key_name(essid, bssid, obj_name, sizeof (obj_name));
435         nlog(LOG_DEBUG, "store_key: obj_name is %s", obj_name);

437         class = (security_mode == DLADM_WLAN_SECMODE_WEP ?
438             DLADM_SECOBJ_CLASS_WEP : DLADM_SECOBJ_CLASS_PSK);
439             DLADM_SECOBJ_CLASS_WEP : DLADM_SECOBJ_CLASS_WPA);
439         if (key_string_to_secobj_value(raw_key, obj_val, &obj_len,
440             class) != 0) {
441                 /* above function logs internally on failure */
442                 nwamd_object_release(ncu_obj);
443                 return (NWAM_ERROR_INTERNAL);
444         }

446         /* we've validated the new key, so remove the old one */
447         status = dladm_unset_secobj(dld_handle, obj_name,
448             DLADM_OPT_ACTIVE | DLADM_OPT_PERSIST);
449         if (status != DLADM_STATUS_OK && status != DLADM_STATUS_NOTFOUND) {
450                 nlog(LOG_ERR, "store_key: could not remove old secure object "
451                     "'%s' for key: %s", obj_name,
452                     dladm_status2str(status, errmsg));
453                 nwamd_object_release(ncu_obj);
454                 return (NWAM_ERROR_INTERNAL);
455         }
```



```
457         /* if we're just deleting the key, then we're done */
458         if (raw_key[0] == '\0') {
459                 nwamd_object_release(ncu_obj);
460                 return (NWAM_SUCCESS);
461         }

463         status = dladm_set_secobj(dld_handle, obj_name, class,
464             obj_val, obj_len,
465             DLADM_OPT_CREATE | DLADM_OPT_PERSIST | DLADM_OPT_ACTIVE);
466         if (status != DLADM_STATUS_OK) {
467                 nlog(LOG_ERR, "store_key: could not create secure object "
468                     "'%s' for key: %s", obj_name,
469                     dladm_status2str(status, errmsg));
470                 nwamd_object_release(ncu_obj);
471                 return (NWAM_ERROR_INTERNAL);
472         }
473         link->nwamd_link_wifi_key = nwamd_wlan_get_key_named(obj_name,
474             security_mode);
475         (void) strlcpy(link->nwamd_link_wifi_keyname, obj_name,
476             sizeof (link->nwamd_link_wifi_keyname));
477         link->nwamd_link_wifi_security_mode = security_mode;
478         if (security_mode == DLADM_WLAN_SECMODE_WEP) {
479                 link->nwamd_link_wifi_key->wk_idx =
480                     (keyslot >= 1 && keyslot <= 4) ? keyslot : 1;
481         }

483         /* If link NCU is offline* or online, (re)connect. */
484         switch (ncu_obj->nwamd_object_state) {
485         case NWAM_STATE_ONLINE:
486                 /* if changing the key of the connected WLAN, reconnect */
487                 if (strcmp(essid, link->nwamd_link_wifi_essid) == 0)
488                         nwamd_object_set_state(NWAM_OBJECT_TYPE_NCU,
489                             ncu_obj->nwamd_object_name, NWAM_STATE_ONLINE,
490                             NWAM_AUX_STATE_LINK_WIFI_CONNECTING);
491                 break;
492         case NWAM_STATE_OFFLINE_TO_ONLINE:
493                 /* if we are waiting for the key, connect */
494                 if (ncu_obj->nwamd_object_aux_state ==
495                     NWAM_AUX_STATE_LINK_WIFI_NEED_KEY)
496                         nwamd_object_set_state(NWAM_OBJECT_TYPE_NCU,
497                             ncu_obj->nwamd_object_name,
498                             NWAM_STATE_OFFLINE_TO_ONLINE,
499                             NWAM_AUX_STATE_LINK_WIFI_CONNECTING);
500                 break;
501         default:
502                 break;
503         }
504         nwamd_object_release(ncu_obj);

506         return (NWAM_SUCCESS);
507 }

509 /*
510  * returns NULL if no key was recovered from libdladm.  Passing in
511  * security mode of 0 means we don't care what key type it is.
512  */
513 dladm_wlan_key_t *
514 nwamd_wlan_get_key_named(const char *name, uint32_t security_mode)
515 {
516         dladm_status_t status;
517         char errmsg[DLADM_STRSIZE];
518         dladm_wlan_key_t *cooked_key;
519         dladm_secobj_class_t class;

521         if (security_mode == DLADM_WLAN_SECMODE_NONE)
522                 return (NULL);
```

```
524         /*
525          * Newly-allocated key must be freed by caller, or by
526          * subsequent call to nwamd_wlan_get_key_named().
527          */
528         if ((cooked_key = malloc(sizeof (dladm_wlan_key_t))) == NULL) {
529                 nlog(LOG_ERR, "nwamd_wlan_get_key_named: malloc failed");
530                 return (NULL);
531         }

533         /*
534          * Set name appropriately to retrieve key for this WLAN.  Note that we
535          * cannot use the actual wk_name buffer size, as it's two times too
536          * large for dladm_get_secobj.
537          */
538         (void) strlcpy(cooked_key->wk_name, name, DLADM_SECOBJ_NAME_MAX);
539         nlog(LOG_DEBUG, "nwamd_wlan_get_key_named: len = %d, object = %s\n",
540             strlen(cooked_key->wk_name), cooked_key->wk_name);
541         cooked_key->wk_len = sizeof (cooked_key->wk_val);
542         cooked_key->wk_idx = 1;

544         /* Try the kernel first, then fall back to persistent storage. */
545         status = dladm_get_secobj(dld_handle, cooked_key->wk_name, &class,
546             cooked_key->wk_val, &cooked_key->wk_len,
547             DLADM_OPT_ACTIVE);
548         if (status != DLADM_STATUS_OK) {
549                 nlog(LOG_DEBUG, "nwamd_wlan_get_key_named: "
550                     "dladm_get_secobj(TEMP) failed: %s",
551                     dladm_status2str(status, errmsg));
552                 status = dladm_get_secobj(dld_handle, cooked_key->wk_name,
553                     &class, cooked_key->wk_val, &cooked_key->wk_len,
554                     DLADM_OPT_PERSIST);
555         }

557         switch (status) {
558         case DLADM_STATUS_OK:
559                 nlog(LOG_DEBUG, "nwamd_wlan_get_key_named: "
560                     "dladm_get_secobj succeeded: len %d", cooked_key->wk_len);
561                 break;
562         case DLADM_STATUS_NOTFOUND:
563                 /*
564                  * We do not want an error in the case that the secobj
565                  * is not found, since we then prompt for it.
566                  */
567                 free(cooked_key);
568                 return (NULL);
569         default:
570                 nlog(LOG_ERR, "nwamd_wlan_get_key_named: could not get key "
571                     "from secure object '%s': %s", cooked_key->wk_name,
572                     dladm_status2str(status, errmsg));
573                 free(cooked_key);
574                 return (NULL);
575         }

577         if (security_mode != 0) {
578                 switch (class) {
579                 case DLADM_SECOBJ_CLASS_WEP:
580                         if (security_mode == DLADM_WLAN_SECMODE_WEP)
581                                 return (cooked_key);
582                         break;
583                 case DLADM_SECOBJ_CLASS_PSK:
584                         if (security_mode == DLADM_WLAN_SECMODE_PSK)
574                 case DLADM_SECOBJ_CLASS_WPA:
575                         if (security_mode == DLADM_WLAN_SECMODE_WPA)
585                                 return (cooked_key);
586                         break;
```



```
587                 default:
588                         /* shouldn't happen */
589                         nlog(LOG_ERR, "nwamd_wlan_get_key: invalid class %d",
590                             class);
591                         break;
592                 }
593                 /* key type mismatch */
594                 nlog(LOG_ERR, "nwamd_wlan_get_key: key type mismatch"
595                     " from secure object '%s'", cooked_key->wk_name);
596                 free(cooked_key);
597                 return (NULL);
598         }

600         cooked_key->wk_class = class;

602 #endif /* ! codereview */
603         return (cooked_key);
604 }

606 static dladm_wlan_key_t *
607 nwamd_wlan_get_key(const char *essid, const char *bssid, uint32_t security_mode)
608 {
609         char keyname[DLADM_SECOBJ_NAME_MAX];

611         nwamd_set_key_name(essid, bssid, keyname, DLADM_SECOBJ_NAME_MAX);

613         return (nwamd_wlan_get_key_named(keyname, security_mode));
614 }

616 /*
617  * Checks if a wireless network can be selected or not.  A wireless network
618  * CANNOT be selected if the NCU is DISABLED, or the NCU is OFFLINE or
619  * ONLINE* and has lower priority than the currently active priority-group.
620  * Called with object lock held.
621  */
622 static boolean_t
623 wireless_selection_possible(nwamd_object_t object)
624 {
625         nwamd_ncu_t *ncu = object->nwamd_object_data;

627         if (ncu->ncu_link.nwamd_link_media != DL_WIFI)
628                 return (B_FALSE);

630         (void) pthread_mutex_lock(&active_ncp_mutex);
631         if (object->nwamd_object_state == NWAM_STATE_DISABLED ||
632             ((object->nwamd_object_state == NWAM_STATE_OFFLINE ||
633             object->nwamd_object_state == NWAM_STATE_ONLINE_TO_OFFLINE) &&
634             ncu->ncu_link.nwamd_link_activation_mode ==
635             NWAM_ACTIVATION_MODE_PRIORITIZED &&
636             (current_ncu_priority_group == INVALID_PRIORITY_GROUP ||
637             ncu->ncu_link.nwamd_link_priority_group >
638             current_ncu_priority_group))) {
639                 (void) pthread_mutex_unlock(&active_ncp_mutex);
640                 return (B_FALSE);
641         }
642         (void) pthread_mutex_unlock(&active_ncp_mutex);

644         return (B_TRUE);
645 }

647 /*
648  * Update the selected and/or connected values for the
649  * scan data.  If these change, we need to trigger a scan
650  * event since the updated values need to be communicated
651  * to the GUI.
652  */
```

```
653 void
654 nwamd_set_selected_connected(nwamd_ncu_t *ncu, boolean_t selected,
655     boolean_t connected)
656 {
657         nwamd_link_t *link = &ncu->ncu_link;
658         nwamd_wifi_scan_t *s = &link->nwamd_link_wifi_scan;
659         int i;
660         boolean_t trigger_scan_event = B_FALSE;

662         for (i = 0; i < s->nwamd_wifi_scan_curr_num; i++) {
663                 if (strcmp(s->nwamd_wifi_scan_curr[i].nww_essid,
664                     link->nwamd_link_wifi_essid) != 0 ||
665                     (link->nwamd_link_wifi_bssid[0] != '\0' &&
666                     strcmp(s->nwamd_wifi_scan_curr[i].nww_bssid,
667                     link->nwamd_link_wifi_bssid) != 0))
668                         continue;
669                 if (selected) {
670                         if (!s->nwamd_wifi_scan_curr[i].nww_selected)
671                                 trigger_scan_event = B_TRUE;
672                         s->nwamd_wifi_scan_curr[i].nww_selected = B_TRUE;
673                 } else {
674                         if (s->nwamd_wifi_scan_curr[i].nww_selected)
675                                 trigger_scan_event = B_TRUE;
676                         s->nwamd_wifi_scan_curr[i].nww_selected = B_FALSE;
677                 }
678                 if (connected) {
679                         if (!s->nwamd_wifi_scan_curr[i].nww_connected)
680                                 trigger_scan_event = B_TRUE;
681                         s->nwamd_wifi_scan_curr[i].nww_connected = B_TRUE;
682                 } else {
683                         if (s->nwamd_wifi_scan_curr[i].nww_connected)
684                                 trigger_scan_event = B_TRUE;
685                         s->nwamd_wifi_scan_curr[i].nww_connected = B_FALSE;
686                 }
687         }

689         if (trigger_scan_event || s->nwamd_wifi_scan_changed) {
690                 nwamd_event_t scan_event = nwamd_event_init_wlan
691                     (ncu->ncu_name, NWAM_EVENT_TYPE_WLAN_SCAN_REPORT, connected,
692                     s->nwamd_wifi_scan_curr, s->nwamd_wifi_scan_curr_num);
693                 if (scan_event != NULL) {
694                         /* Avoid sending same scan data multiple times */
695                         s->nwamd_wifi_scan_changed = B_FALSE;
696                         nwamd_event_enqueue(scan_event);
697                 }
698         }
699 }

701 /*
702  * Callback used on each known WLAN - if the BSSID is matched, set
703  * the ESSID of the hidden WLAN to the known WLAN name.
704  */
705 static int
706 find_bssid_cb(nwam_known_wlan_handle_t kwh, void *data)
707 {
708         nwamd_link_t *link = data;
709         nwam_error_t err;
710         nwam_value_t bssidval;
711         char **bssids, *name;
712         uint_t num_bssids, i;

714         if ((err = nwam_known_wlan_get_prop_value(kwh,
715             NWAM_KNOWN_WLAN_PROP_BSSIDS, &bssidval)) != NWAM_SUCCESS) {
716                 nlog(LOG_ERR, "find_bssid_cb: nwam_known_wlan_get_prop: %s",
717                     nwam_strerror(err));
718                 return (0);
```



```
719         }
720         if ((err = nwam_value_get_string_array(bssidval, &bssids, &num_bssids))
721             != NWAM_SUCCESS) {
722                 nlog(LOG_ERR, "find_bssid_cb: nwam_value_get_string_array: %s",
723                     nwam_strerror(err));
724                 nwam_value_free(bssidval);
725                 return (0);
726         }
727         for (i = 0; i < num_bssids; i++) {
728                 if (strcmp(bssids[i], link->nwamd_link_wifi_bssid) == 0) {
729                         if ((err = nwam_known_wlan_get_name(kwh, &name))
730                             != NWAM_SUCCESS) {
731                                 nlog(LOG_ERR, "find_bssid_cb: "
732                                     "nwam_known_wlan_get_name: %s",
733                                     nwam_strerror(err));
734                                 continue;
735                         }
736                         (void) strlcpy(link->nwamd_link_wifi_essid, name,
737                             sizeof (link->nwamd_link_wifi_essid));
738                         free(name);
739                         nwam_value_free(bssidval);
740                         /* Found ESSID for BSSID so terminate walk */
741                         return (1);
742                 }
743         }
744         nwam_value_free(bssidval);

746         return (0);
747 }

749 /*
750  * We may have encountered a BSSID for a hidden WLAN before and as a result
751  * may have a known WLAN entry with this BSSID.  Walk known WLANs, searching
752  * for a BSSID match.  Called with object lock held.
753  */
754 static void
755 check_if_hidden_wlan_was_visited(nwamd_link_t *link)
756 {
757         (void) nwam_walk_known_wlans(find_bssid_cb, link,
758             NWAM_FLAG_KNOWN_WLAN_WALK_PRIORITY_ORDER, NULL);
759 }

761 nwam_error_t
762 nwamd_wlan_select(const char *linkname, const char *essid, const char *bssid,
763     uint32_t security_mode, boolean_t add_to_known_wlans)
764 {
765         nwamd_object_t ncu_obj;
766         nwamd_ncu_t *ncu;
767         nwamd_link_t *link;
768         char key[DLADM_STRSIZE];
769         boolean_t found_old_key = B_FALSE, found_key = B_FALSE;

771         if ((ncu_obj = nwamd_ncu_object_find(NWAM_NCU_TYPE_LINK, linkname))
772             == NULL) {
773                 nlog(LOG_ERR, "nwamd_wlan_select: could not find object  "
774                     "for link %s", linkname);
775                 return (NWAM_ENTITY_NOT_FOUND);
776         }
777         ncu = ncu_obj->nwamd_object_data;
778         link = &ncu->ncu_link;

780         /*
781          * If wireless selection is not possible because of the current
782          * state or priority-group, then stop.
783          */
784         if (!wireless_selection_possible(ncu_obj)) {
```

```
785                 nwamd_object_release(ncu_obj);
786                 return (NWAM_ENTITY_INVALID_STATE);
787         }
789         /* unset selected, connected flag for previously connected wlan */
790         nwamd_set_selected_connected(ncu, B_FALSE, B_FALSE);
792         /* Disconnect to allow new selection to go ahead */
793         (void) dladm_wlan_disconnect(dld_handle, link->nwamd_link_id);
795         (void) strlcpy(link->nwamd_link_wifi_essid, essid,
796             sizeof (link->nwamd_link_wifi_essid));
797         (void) strlcpy(link->nwamd_link_wifi_bssid, bssid,
798             sizeof (link->nwamd_link_wifi_bssid));
799         link->nwamd_link_wifi_security_mode = security_mode;
800         link->nwamd_link_wifi_add_to_known_wlans = add_to_known_wlans;
802         /* If this is a hidden wlan, then essid is empty */
803         if (link->nwamd_link_wifi_essid[0] == '\0')
804                 check_if_hidden_wlan_was_visited(link);
806         /* set selected flag for newly-selected WLAN */
807         nwamd_set_selected_connected(ncu, B_TRUE, B_FALSE);
809         /* does this WLAN require a key? If so go to NEED_KEY */
810         if (NEED_ENC(link->nwamd_link_wifi_security_mode)) {
811                 /*
812                  * First, if a key name may have been specified for a
813                  * known WLAN.  If so, use it.  Otherwise, try both the
814                  * new nwamd key name format (ESSID) and old (ESSID/BSSID).
815                  * The user may have set the key without adding a known WLAN,
816                  * so we need to try all these options to save going to
817                  * NEED_KEY state.
818                  */
819                 if (known_wlan_get_keyname(link->nwamd_link_wifi_essid,
820                     link->nwamd_link_wifi_keyname) == NWAM_SUCCESS &&
821                     (link->nwamd_link_wifi_key = nwamd_wlan_get_key_named
822                     (link->nwamd_link_wifi_keyname,
823                     link->nwamd_link_wifi_security_mode)) != NULL) {
824                         (void) known_wlan_get_keyslot
825                             (link->nwamd_link_wifi_essid,
826                             &link->nwamd_link_wifi_key->wk_idx);
827                         nlog(LOG_DEBUG, "nwamd_wlan_select: got known WLAN "
828                             "key %s, slot %d", link->nwamd_link_wifi_keyname,
829                             link->nwamd_link_wifi_key->wk_idx);
830                         found_key = B_TRUE;
831                 } else if ((link->nwamd_link_wifi_key = nwamd_wlan_get_key
832                     (link->nwamd_link_wifi_essid, NULL,
833                     link->nwamd_link_wifi_security_mode)) != NULL) {
834                         nwamd_set_key_name(link->nwamd_link_wifi_essid, NULL,
835                             link->nwamd_link_wifi_keyname,
836                             DLADM_SECOBJ_NAME_MAX);
837                         nlog(LOG_DEBUG, "nwamd_wlan_select: got WLAN key %s",
838                             link->nwamd_link_wifi_keyname);
839                         found_key = B_TRUE;
840                 } else if ((link->nwamd_link_wifi_key = nwamd_wlan_get_key
841                     (link->nwamd_link_wifi_essid, link->nwamd_link_wifi_bssid,
842                     link->nwamd_link_wifi_security_mode)) != NULL) {
843                         /*
844                          * Found old key format - prepare to save
845                          * it as new ESSID-only key, but don't
846                          * do it until we're released the object
847                          * lock (since nwamd_wlan_set_key()
848                          * takes the object lock).
849                          */
850                         (void) strlcpy(key,
```



```
851                             (char *)link->nwamd_link_wifi_key->wk_val,
852                             link->nwamd_link_wifi_key->wk_len + 1);
853                         found_old_key = B_TRUE;
854                         found_key = B_TRUE;
855                         nwamd_set_key_name(link->nwamd_link_wifi_essid, NULL,
856                             link->nwamd_link_wifi_keyname,
857                             DLADM_SECOBJ_NAME_MAX);
858                         nlog(LOG_DEBUG, "nwamd_wlan_select: got old format "
859                             "WLAN key, converting to %s",
860                             link->nwamd_link_wifi_keyname);
861                 } else {
862                         nlog(LOG_ERR, "nwamd_wlan_select: could not "
863                             "find key for WLAN '%s'",
864                             link->nwamd_link_wifi_essid);
865                 }
866         } else {
867                 free(link->nwamd_link_wifi_key);
868                 link->nwamd_link_wifi_key = NULL;
869                 link->nwamd_link_wifi_keyname[0] = '\0';
870         }
872         if (NEED_ENC(link->nwamd_link_wifi_security_mode) && !found_key) {
873                 nwamd_object_set_state(NWAM_OBJECT_TYPE_NCU,
874                     ncu_obj->nwamd_object_name,
875                     NWAM_STATE_OFFLINE_TO_ONLINE,
876                     NWAM_AUX_STATE_LINK_WIFI_NEED_KEY);
877         } else {
878                 nwamd_object_set_state(NWAM_OBJECT_TYPE_NCU,
879                     ncu_obj->nwamd_object_name, NWAM_STATE_OFFLINE_TO_ONLINE,
880                     NWAM_AUX_STATE_LINK_WIFI_CONNECTING);
881         }
882         nwamd_object_release(ncu_obj);
884         if (found_old_key) {
885                 (void) nwamd_wlan_set_key(linkname, essid, NULL, security_mode,
886                     1, key);
887         }
888         return (NWAM_SUCCESS);
889 }
891 /*
892  * See if BSSID is in visited list of BSSIDs for known WLAN. Used for
893  * strict BSSID matching (depends on wireless_strict_bssid property value).
894  */
895 static boolean_t
896 bssid_match(nwam_known_wlan_handle_t kwh, const char *bssid)
897 {
898         nwam_value_t bssidsval;
899         nwam_error_t err;
900         char **bssids;
901         uint_t nelem, i;
902         boolean_t found = B_FALSE;
904         if ((err = nwam_known_wlan_get_prop_value(kwh,
905             NWAM_KNOWN_WLAN_PROP_BSSIDS, &bssidsval)) != NWAM_SUCCESS) {
906                 nlog(LOG_ERR, "bssid_match: %s", nwam_strerror(err));
907                 return (B_FALSE);
908         }
909         if ((err = nwam_value_get_string_array(bssidsval, &bssids, &nelem))
910             != NWAM_SUCCESS) {
911                 nwam_value_free(bssidsval);
912                 return (B_FALSE);
913         }
914         for (i = 0; i < nelem; i++) {
915                 if (strcmp(bssid, bssids[i]) == 0) {
916                         found = B_TRUE;
```

```
917                         break;
918                 }
919         }
920         nwam_value_free(bssidsval);
921
922         return (found);
923 }
924
925 /* Find most prioritized AP with strongest signal in scan data. */
926 static int
927 find_best_wlan_cb(nwam_known_wlan_handle_t kwh, void *data)
928 {
929         nwamd_ncu_t *ncu = data;
930         nwamd_link_t *link = &ncu->ncu_link;
931         nwamd_wifi_scan_t *s = &link->nwamd_link_wifi_scan;
932         nwam_error_t err;
933         char *name = NULL;
934         int i;
935         dladm_wlan_strength_t curr_strength = 0;
936         dladm_wlan_strength_t max_strength = 0;
937         boolean_t found = B_FALSE;
938
939         if ((err = nwam_known_wlan_get_name(kwh, &name)) != NWAM_SUCCESS) {
940                 nlog(LOG_ERR, "find_best_wlan_cb: could not look up name: %s",
941                     nwam_strerror(err));
942                 return (0);
943         }
944
945         if (link->nwamd_link_wifi_connected) {
946                 (void) dladm_wlan_str2strength
947                     (link->nwamd_link_wifi_signal_strength, &curr_strength);
948         }
949
950         /*
951          * If we're >= scan level, don't pick another Known WLAN if still
952          * connected (even if a Known WLAN with higher priority is available).
953          * If the user wants to connect to a different Known WLAN, it can be
954          * done from the GUI or select-wifi subcommand of nwamadm(1M).
955          */
956         if (curr_strength >= wireless_scan_level &&
957             link->nwamd_link_wifi_connected) {
958                 free(name);
959                 return (1);
960         }
961
962         for (i = 0; i < s->nwamd_wifi_scan_curr_num; i++) {
963                 nwam_wlan_t *cur_wlan = &(s->nwamd_wifi_scan_curr[i]);
964                 boolean_t b_match = bssid_match(kwh, cur_wlan->nww_bssid);
965
966                 /*
967                  * We need to either match the scanned essid, or in the case
968                  * where the essid was not broadcast, match the scanned bssid.
969                  */
970                 if (strcmp(cur_wlan->nww_essid, name) != 0 &&
971                     !(cur_wlan->nww_essid[0] == '\0' && b_match))
972                         continue;
973                 /*
974                  * If wireless_strict_bssid is specified, need to match
975                  * BSSID too.
976                  */
977                 if (wireless_strict_bssid && !b_match)
978                         continue;
979                 /*
980                  * Found a match. Since we walk known WLANs in
981                  * priority order, it's guaranteed to be the
982                  * most prioritized. It may not be the strongest though -
```



```
983                  * we continue the walk and record the strength along
984                  * with the ESSID and BSSID, so that if we encounter
985                  * another AP with the same ESSID but a higher signal strength,
986                  * we will choose it - but only if the currently-connected
987                  * WLAN is at or below wireless_scan_level.
988                  */
989                 (void) dladm_wlan_str2strength
990                     (cur_wlan->nww_signal_strength, &curr_strength);
991
992                 if (curr_strength > max_strength) {
993                         (void) strlcpy(link->nwamd_link_wifi_essid,
994                             cur_wlan->nww_essid,
995                             sizeof (link->nwamd_link_wifi_essid));
996                         /*
997                          * Set BSSID if wireless_strict_bssid is specified or
998                          * if this is a hidden WLAN.  Store the BSSID here and
999                          * then later determine the hidden WLAN's name in the
1000                         * connect thread.
1001                         */
1002                        if (wireless_strict_bssid ||
1003                            cur_wlan->nww_essid[0] == '\0') {
1004                                (void) strlcpy(link->nwamd_link_wifi_bssid,
1005                                    cur_wlan->nww_bssid,
1006                                    sizeof (link->nwamd_link_wifi_bssid));
1007                        }
1008                        (void) strlcpy(link->nwamd_link_wifi_signal_strength,
1009                            cur_wlan->nww_signal_strength,
1010                            sizeof (link->nwamd_link_wifi_signal_strength));
1011                        link->nwamd_link_wifi_security_mode =
1012                            cur_wlan->nww_security_mode;
1013                        found = B_TRUE;
1014                }
1015                (void) dladm_wlan_str2strength
1016                    (link->nwamd_link_wifi_signal_strength, &max_strength);
1017        }
1018        free(name);
1019        return (found ? 1 : 0);
1020 }
1021
1022 static boolean_t
1023 nwamd_find_known_wlan(nwamd_object_t ncu_obj)
1024 {
1025         nwamd_ncu_t *ncu = ncu_obj->nwamd_object_data;
1026         int ret;
1027
1028         /*
1029          * Walk known WLANs, finding lowest priority (preferred) WLAN
1030          * in our scan results.
1031          */
1032         (void) nwam_walk_known_wlans(find_best_wlan_cb, ncu,
1033             NWAM_FLAG_KNOWN_WLAN_WALK_PRIORITY_ORDER, &ret);
1034
1035         return (ret == 1);
1036 }
1037
1038 /*
1039  * WLAN scan code for WIFI link NCUs.
1040  */
1041
1042 /* Create periodic scan event for object.  Called with object lock held. */
1043 void
1044 nwamd_ncu_create_periodic_scan_event(nwamd_object_t ncu_obj)
1045 {
1046         nwamd_event_t scan_event;
1047
1048         if (wireless_scan_interval == 0) {
```

```
1049                 nlog(LOG_DEBUG, "nwamd_ncu_create_periodic_scan_event: "
1050                     "wireless_scan_interval set to 0 so no periodic scanning");
1051                 return;
1052         }
1053         scan_event = nwamd_event_init(NWAM_EVENT_TYPE_PERIODIC_SCAN,
1054             NWAM_OBJECT_TYPE_NCU, 0, ncu_obj->nwamd_object_name);
1055         if (scan_event != NULL) {
1056                 nwamd_event_enqueue_timed(scan_event,
1057                     wireless_scan_interval > WIRELESS_SCAN_INTERVAL_MIN ?
1058                     wireless_scan_interval : WIRELESS_SCAN_INTERVAL_MIN);
1059         }
1060 }

1062 /* Handle periodic scan event (which puts link into WIFI_INIT state */
1063 void
1064 nwamd_ncu_handle_periodic_scan_event(nwamd_event_t event)
1065 {
1066         nwamd_object_t ncu_obj;
1067         nwamd_ncu_t *ncu;

1069         ncu_obj = nwamd_object_find(NWAM_OBJECT_TYPE_NCU,
1070             event->event_object);
1071         if (ncu_obj == NULL) {
1072                 nlog(LOG_ERR, "nwamd_ncu_handle_periodic_scan_event: "
1073                     "no object %s", event->event_object);
1074                 return;
1075         }
1076         ncu = ncu_obj->nwamd_object_data;

1078         /* Only rescan if state is offline* or online */
1079         nlog(LOG_DEBUG, "nwamd_ncu_handle_periodic_scan_event: doing rescan..");

1081         if (ncu_obj->nwamd_object_state == NWAM_STATE_OFFLINE_TO_ONLINE ||
1082             ncu_obj->nwamd_object_state == NWAM_STATE_ONLINE) {
1083                 /* rescan, then create periodic scan event */
1084                 (void) nwamd_wlan_scan(ncu->ncu_name);
1085                 nwamd_ncu_create_periodic_scan_event(ncu_obj);
1086         }
1087         nwamd_object_release(ncu_obj);
1088 }

1090 static boolean_t
1091 get_scan_results(void *arg, dladm_wlan_attr_t *attrp)
1092 {
1093         nwamd_wifi_scan_t *s = arg;
1094         const char *linkname = s->nwamd_wifi_scan_link;
1095         char essid_name[DLADM_STRSIZE];
1096         char bssid_name[DLADM_STRSIZE];
1097         char strength[DLADM_STRSIZE];
1098         uint_t i, index = 0;
1099         boolean_t found = B_FALSE;

1101         (void) dladm_wlan_essid2str(&attrp->wa_essid, essid_name);
1102         (void) dladm_wlan_bssid2str(&attrp->wa_bssid, bssid_name);
1103         (void) dladm_wlan_strength2str(&attrp->wa_strength, strength);

1105         index = s->nwamd_wifi_scan_curr_num;
1106         if (index == NWAMD_MAX_NUM_WLANS) {
1107                 nlog(LOG_ERR, "get_scan_results: truncating WLAN scan results "
1108                     "for link %s: ommiting (%s, %s)", linkname, essid_name,
1109                     bssid_name);
1110                 return (B_TRUE);
1111         }

1113         (void) strlcpy(s->nwamd_wifi_scan_curr[index].nww_essid, essid_name,
1114             sizeof (s->nwamd_wifi_scan_curr[index].nww_essid));
```



```
1115         (void) strlcpy(s->nwamd_wifi_scan_curr[index].nww_bssid, bssid_name,
1116             sizeof (s->nwamd_wifi_scan_curr[index].nww_bssid));
1117         (void) strlcpy(s->nwamd_wifi_scan_curr[index].nww_signal_strength,
1118             strength,
1119             sizeof (s->nwamd_wifi_scan_curr[index].nww_signal_strength));
1120         s->nwamd_wifi_scan_curr[index].nww_security_mode = attrp->wa_secmode;
1121         s->nwamd_wifi_scan_curr[index].nww_speed = attrp->wa_speed;
1122         s->nwamd_wifi_scan_curr[index].nww_channel = attrp->wa_channel;
1123         s->nwamd_wifi_scan_curr[index].nww_bsstype = attrp->wa_bsstype;

1125         /*
1126          * We fill in actual values for selected/connected/key later when we
1127          * reacquire the object lock.
1128          */
1129         s->nwamd_wifi_scan_curr[index].nww_selected = B_FALSE;
1130         s->nwamd_wifi_scan_curr[index].nww_connected = B_FALSE;
1131         s->nwamd_wifi_scan_curr[index].nww_have_key = B_FALSE;
1132         s->nwamd_wifi_scan_curr[index].nww_keyindex = 1;
1133         s->nwamd_wifi_scan_curr_num++;

1135         /* Check if this AP was in previous scan results */
1136         for (i = 0; i < s->nwamd_wifi_scan_last_num; i++) {
1137                 found = (strcmp(s->nwamd_wifi_scan_last[i].nww_essid,
1138                     essid_name) == 0 &&
1139                     strcmp(s->nwamd_wifi_scan_last[i].nww_bssid,
1140                     bssid_name) == 0);
1141                 if (found)
1142                         break;
1143         }
1144         if (!found)
1145                 s->nwamd_wifi_scan_changed = B_TRUE;

1147         nlog(LOG_DEBUG, "get_scan_results(%s, %d): ESSID %s, BSSID %s",
1148             linkname, index, essid_name, bssid_name);

1150         return (B_TRUE);
1151 }

1153 /*
1154  * Check if we're connected to the expected WLAN, or in the case of autoconf
1155  * record the WLAN we're connected to.
1156  */
1157 boolean_t
1158 nwamd_wlan_connected(nwamd_object_t ncu_obj)
1159 {
1160         nwamd_ncu_t *ncu = ncu_obj->nwamd_object_data;
1161         nwamd_link_t *link = &ncu->ncu_link;
1162         dladm_wlan_linkattr_t attr;
1163         char essid[DLADM_STRSIZE];
1164         char bssid[DLADM_STRSIZE];
1165         boolean_t connected = B_FALSE;
1166         int retries = 0;

1168         /*
1169          * This is awful, but some wireless drivers
1170          * (particularly 'ath') will erroneously report
1171          * "disconnected" if queried right after a scan.  If we
1172          * see 'down' reported here, we retry a few times to
1173          * make sure it's really down.
1174          */
1175         while (retries++ < 4) {
1176                 if (dladm_wlan_get_linkattr(dld_handle, link->nwamd_link_id,
1177                     &attr) != DLADM_STATUS_OK) {
1178                         attr.la_status = DLADM_WLAN_LINK_DISCONNECTED;
1179                 } else if (attr.la_status == DLADM_WLAN_LINK_CONNECTED) {
1180                         break;
```

```
1181                }
1182        }

1184        if (attr.la_status == DLADM_WLAN_LINK_CONNECTED) {
1185                (void) dladm_wlan_essid2str(&attr.la_wlan_attr.wa_essid, essid);
1186                (void) dladm_wlan_bssid2str(&attr.la_wlan_attr.wa_bssid, bssid);
1187                connected = B_TRUE;
1188                nlog(LOG_DEBUG, "nwamd_wlan_connected: %s connected to %s %s",
1189                    ncu->ncu_name, essid, bssid);
1190        } else {
1191                return (B_FALSE);
1192        }
1193        /*
1194         * If we're using autoconf,  we have no control over what we connect to,
1195         * so rather than verifying ESSSID, simply record ESSID/BSSID.
1196         */
1197        if (link->nwamd_link_wifi_autoconf) {
1198                (void) strlcpy(link->nwamd_link_wifi_essid, essid,
1199                    sizeof (link->nwamd_link_wifi_essid));
1200                (void) strlcpy(link->nwamd_link_wifi_bssid, bssid,
1201                    sizeof (link->nwamd_link_wifi_bssid));
1202        }
1203        /*
1204         * Are we connected to expected WLAN? Note:
1205         * we'd like to verify BSSID, but we cannot due to CR 6772510.
1206         */
1207        if (strcmp(essid, link->nwamd_link_wifi_essid) == 0) {
1208                /* Update connected signal strength */
1209                (void) dladm_wlan_strength2str(&attr.la_wlan_attr.wa_strength,
1210                    link->nwamd_link_wifi_signal_strength);

1212                /* Store current BSSID */
1213                (void) strlcpy(link->nwamd_link_wifi_bssid, bssid,
1214                    sizeof (link->nwamd_link_wifi_bssid));

1216                if (attr.la_wlan_attr.wa_strength < wireless_scan_level) {
1217                        /*
1218                         * We're connected, but we've dropped below
1219                         * scan threshold.  Initiate a scan.
1220                         */
1221                        nlog(LOG_DEBUG, "nwamd_wlan_connected: "
1222                            "connected but signal under threshold...");
1223                        (void) nwamd_wlan_scan(ncu->ncu_name);
1224                }
1225                return (connected);
1226        } else if (strlen(essid) == 0) {
1227                /*
1228                 * For hidden WLANs, no ESSID is specified, so we cannot verify
1229                 * WLAN name.
1230                 */
1231                nlog(LOG_DEBUG,
1232                    "nwamd_wlan_connected: connected to hidden WLAN, cannot "
1233                    "verify connection details");
1234                return (connected);
1235        } else {
1236                (void) nlog(LOG_ERR,
1237                    "nwamd_wlan_connected: wrong AP on %s; expected %s %s",
1238                    ncu->ncu_name, link->nwamd_link_wifi_essid,
1239                    link->nwamd_link_wifi_bssid);
1240                (void) dladm_wlan_disconnect(dld_handle, link->nwamd_link_id);
1241                link->nwamd_link_wifi_connected = B_FALSE;
1242                return (B_FALSE);
1243        }
1244 }

1246 /*
```



```
1247  * WLAN scan thread. Called with the per-link WiFi mutex held.
1248  */
1249 static void *
1250 wlan_scan_thread(void *arg)
1251 {
1252         char *linkname = arg;
1253         nwamd_object_t ncu_obj;
1254         nwamd_ncu_t *ncu;
1255         nwamd_link_t *link;
1256         dladm_status_t status;
1257         char essid[DLADM_STRSIZE];
1258         char bssid[DLADM_STRSIZE];
1259         uint32_t now, link_id;
1260         nwamd_wifi_scan_t s;
1261         int i;

1263         if ((ncu_obj = nwamd_ncu_object_find(NWAM_NCU_TYPE_LINK, linkname))
1264             == NULL) {
1265                 nlog(LOG_ERR, "wlan_scan_thread: could not find object  "
1266                     "for link %s", linkname);
1267                 free(linkname);
1268                 return (NULL);
1269         }

1271         ncu = ncu_obj->nwamd_object_data;
1272         link = &ncu->ncu_link;

1274         /*
1275          * It is possible multiple scan threads have queued up waiting for the
1276          * object lock.  We try to prevent excessive scanning by limiting the
1277          * interval between scans to WIRELESS_SCAN_REQUESTED_INTERVAL_MIN sec.
1278          */
1279         now = NSEC_TO_SEC(gethrtime());
1280         if ((now - link->nwamd_link_wifi_scan.nwamd_wifi_scan_last_time) <
1281             WIRELESS_SCAN_REQUESTED_INTERVAL_MIN) {
1282                 nlog(LOG_DEBUG, "wlan_scan_thread: last scan for %s "
1283                     "was < %d sec ago, ignoring scan request",
1284                     linkname, WIRELESS_SCAN_REQUESTED_INTERVAL_MIN);
1285                 nwamd_object_release(ncu_obj);
1286                 free(linkname);
1287                 return (NULL);
1288         }

1290         /*
1291          * Prepare scan data - copy link name and copy previous "current"
1292          * scan results from the nwamd_link_t to the last scan results for
1293          * the next scan so that we can compare results to find if things
1294          * have changed since last time.
1295          */
1296         (void) bzero(&s, sizeof (nwamd_wifi_scan_t));
1297         (void) strlcpy(s.nwamd_wifi_scan_link, ncu->ncu_name,
1298             sizeof (s.nwamd_wifi_scan_link));
1299         s.nwamd_wifi_scan_last_num =
1300             link->nwamd_link_wifi_scan.nwamd_wifi_scan_curr_num;
1301         if (s.nwamd_wifi_scan_last_num > 0) {
1302                 (void) memcpy(s.nwamd_wifi_scan_last,
1303                     link->nwamd_link_wifi_scan.nwamd_wifi_scan_curr,
1304                     s.nwamd_wifi_scan_last_num * sizeof (nwam_wlan_t));
1305         }
1306         link_id = link->nwamd_link_id;
1307         nwamd_object_release(ncu_obj);

1309         nlog(LOG_DEBUG, "wlan_scan_thread: initiating scan on %s",
1310             s.nwamd_wifi_scan_link);

1312         scanconnect_entry();
```

```
1313         status = dladm_wlan_scan(dld_handle, link_id, &s, get_scan_results);
1314         s.nwamd_wifi_scan_last_time = NSEC_TO_SEC(gethrtime());
1315         if (!s.nwamd_wifi_scan_changed) {
1316                 /* Scan may have lost WLANs, if so this qualifies as change */
1317                 s.nwamd_wifi_scan_changed = (s.nwamd_wifi_scan_curr_num !=
1318                     s.nwamd_wifi_scan_last_num);
1319         }
1320         scanconnect_exit();

1322         if (status != DLADM_STATUS_OK) {
1323                 nlog(LOG_ERR, "wlan_scan_thread: cannot scan link %s",
1324                     s.nwamd_wifi_scan_link);
1325                 free(linkname);
1326                 return (NULL);
1327         }

1329         if ((ncu_obj = nwamd_ncu_object_find(NWAM_NCU_TYPE_LINK, linkname))
1330             == NULL) {
1331                 nlog(LOG_ERR, "wlan_scan_thread: could not find object  "
1332                     "for link %s after doing scan", linkname);
1333                 free(linkname);
1334                 return (NULL);
1335         }
1336         ncu = ncu_obj->nwamd_object_data;
1337         link = &ncu->ncu_link;

1339         /* For new scan data, add key info from known WLANs */
1340         for (i = 0; i < s.nwamd_wifi_scan_curr_num; i++) {
1341                 if (NEED_ENC(s.nwamd_wifi_scan_curr[i].nww_security_mode)) {
1342                         char keyname[NWAM_MAX_VALUE_LEN];
1343                         dladm_wlan_key_t *key = NULL;

1345                         if (known_wlan_get_keyname
1346                             (s.nwamd_wifi_scan_curr[i].nww_essid, keyname)
1347                             == NWAM_SUCCESS &&
1348                             (key = nwamd_wlan_get_key_named(keyname,
1349                             s.nwamd_wifi_scan_curr[i].nww_security_mode))
1350                             != NULL) {
1351                                 s.nwamd_wifi_scan_curr[i].nww_have_key =
1352                                     B_TRUE;
1353                                 s.nwamd_wifi_scan_curr[i].nww_keyindex =
1354                                     s.nwamd_wifi_scan_curr[i].
1355                                     nww_security_mode ==
1356                                     DLADM_WLAN_SECMODE_WEP ?
1357                                     key->wk_idx : 1;
1358                                 free(key);
1359                         }
1360                 }
1361         }
1362         /* Copy scan data into nwamd_link_t */
1363         link->nwamd_link_wifi_scan = s;
1364         /* Set selected, connected and send scan event if we've got new data */
1365         nwamd_set_selected_connected(ncu,
1366             link->nwamd_link_wifi_essid[0] != '\0',
1367             link->nwamd_link_wifi_connected);

1369         /*
1370          * If wireless selection is not possible because of the current
1371          * state or priority-group, then this was just a scan request.
1372          * Nothing else to do.
1373          */
1374         if (!wireless_selection_possible(ncu_obj)) {
1375                 nwamd_object_release(ncu_obj);
1376                 free(linkname);
1377                 return (NULL);
1378         }
```



```
1380         /*
1381          * Check if WLAN is on our known WLAN list. If no
1382          * previously-visited WLANs are found in scan data, set
1383          * new state to NEED_SELECTION (provided we're not currently
1384          * connected, as can be the case during a periodic scan or
1385          * monitor-triggered scan where the signal strength recovers.
1386          */
1387         if (!nwamd_find_known_wlan(ncu_obj)) {
1388                 if (!nwamd_wlan_connected(ncu_obj)) {
1389                         if (link->nwamd_link_wifi_connected) {
1390                                 nlog(LOG_DEBUG, "wlan_scan_thread: "
1391                                     "unexpected disconnect after scan");
1392                                 nwamd_object_set_state(NWAM_OBJECT_TYPE_NCU,
1393                                     ncu_obj->nwamd_object_name,
1394                                     NWAM_STATE_ONLINE_TO_OFFLINE,
1395                                     NWAM_AUX_STATE_DOWN);
1396                         } else {
1397                                 nlog(LOG_DEBUG, "wlan_scan_thread: "
1398                                     "no known WLANs - ask user");
1399                                 nwamd_object_set_state(NWAM_OBJECT_TYPE_NCU,
1400                                     ncu_obj->nwamd_object_name,
1401                                     NWAM_STATE_OFFLINE_TO_ONLINE,
1402                                     NWAM_AUX_STATE_LINK_WIFI_NEED_SELECTION);
1403                         }
1404                 } else {
1405                         /* still connected. if not online, change to online */
1406                         nlog(LOG_DEBUG, "wlan_scan_thread: still connected to "
1407                             "%s %s", link->nwamd_link_wifi_essid,
1408                             link->nwamd_link_wifi_bssid);
1409                         if (ncu_obj->nwamd_object_state != NWAM_STATE_ONLINE) {
1410                                 nwamd_object_set_state(NWAM_OBJECT_TYPE_NCU,
1411                                     ncu_obj->nwamd_object_name,
1412                                     NWAM_STATE_OFFLINE_TO_ONLINE,
1413                                     NWAM_AUX_STATE_UP);
1414                         }
1415                 }
1416                 nwamd_object_release(ncu_obj);

1418         } else {
1419                 nlog(LOG_DEBUG, "wlan_scan_thread: found known WLAN %s %s",
1420                     link->nwamd_link_wifi_essid, link->nwamd_link_wifi_bssid);

1422                 if (!nwamd_wlan_connected(ncu_obj)) {
1423                         /* Copy selected ESSID/BSSID, unlock, call select */
1424                         (void) strlcpy(essid, link->nwamd_link_wifi_essid,
1425                             sizeof (essid));
1426                         (void) strlcpy(bssid, link->nwamd_link_wifi_bssid,
1427                             sizeof (bssid));
1428                         nwamd_object_release(ncu_obj);
1429                         (void) nwamd_wlan_select(linkname, essid, bssid,
1430                             link->nwamd_link_wifi_security_mode, B_TRUE);
1431                 } else {
1432                         /* still connected.  if not online, change to online */
1433                         nlog(LOG_DEBUG, "wlan_scan_thread: still connected to "
1434                             "known WLAN %s %s", link->nwamd_link_wifi_essid,
1435                             link->nwamd_link_wifi_bssid);
1436                         if (ncu_obj->nwamd_object_state != NWAM_STATE_ONLINE) {
1437                                 nwamd_object_set_state(NWAM_OBJECT_TYPE_NCU,
1438                                     ncu_obj->nwamd_object_name,
1439                                     NWAM_STATE_OFFLINE_TO_ONLINE,
1440                                     NWAM_AUX_STATE_UP);
1441                         }
1442                         nwamd_object_release(ncu_obj);
1443                 }
1444         }
```

```
1445         free(linkname);
1446         return (NULL);
1447 }

1449 nwam_error_t
1450 nwamd_wlan_scan(const char *linkname)
1451 {
1452         pthread_t wifi_thread;
1453         char *link = strdup(linkname);

1455         if (link == NULL) {
1456                 nlog(LOG_ERR, "nwamd_wlan_scan: out of memory");
1457                 return (NWAM_NO_MEMORY);
1458         }

1460         nlog(LOG_DEBUG, "nwamd_wlan_scan: WLAN scan for %s",
1461             link);

1463         if (pthread_create(&wifi_thread, NULL, wlan_scan_thread,
1464             link) != 0) {
1465                 nlog(LOG_ERR, "nwamd_wlan_scan: could not start scan");
1466                 free(link);
1467                 return (NWAM_ERROR_INTERNAL);
1468         }
1469         /* detach thread so that it doesn't become a zombie */
1470         (void) pthread_detach(wifi_thread);
1471         return (NWAM_SUCCESS);
1472 }

1474 /*
1475  * WLAN connection code.
1476  */

1478 static dladm_status_t
1479 do_connect(uint32_t link_id, dladm_wlan_attr_t *attrp, dladm_wlan_key_t *key,
1480     uint_t keycount, uint_t flags)
1481 {
1482         dladm_status_t status;
1483         char errmsg[DLADM_STRSIZE];

1485         scanconnect_entry();
1486         status = dladm_wlan_connect(dld_handle, link_id, attrp,
1487             DLADM_WLAN_CONNECT_TIMEOUT_DEFAULT, key, keycount, flags, NULL);
 591             DLADM_WLAN_CONNECT_TIMEOUT_DEFAULT, key, keycount, flags);
1488         scanconnect_exit();

1490         nlog(LOG_DEBUG, "nwamd_do_connect: dladm_wlan_connect returned %s",
1491             dladm_status2str(status, errmsg));

1493         return (status);
1494 }

1496 static void *
1497 wlan_connect_thread(void *arg)
1498 {
1499         char *linkname = arg;
1500         nwamd_object_t ncu_obj;
1501         nwamd_ncu_t *ncu;
1502         nwamd_link_t *link;
1503         nwam_error_t err;
1504         uint_t  keycount;
1505         uint32_t link_id;
1506         dladm_wlan_key_t *key = NULL;
1507         dladm_wlan_attr_t attr;
1508         dladm_status_t status;
1509         boolean_t autoconf = B_FALSE;
```



```
1511         if ((ncu_obj = nwamd_ncu_object_find(NWAM_NCU_TYPE_LINK, linkname))
1512             == NULL) {
1513                 nlog(LOG_ERR, "wlan_connect_thread: could not find object  "
1514                     "for link %s", linkname);
1515                 free(linkname);
1516                 return (NULL);
1517         }

1519         ncu = ncu_obj->nwamd_object_data;
1520         link = &ncu->ncu_link;

1522         if (!wireless_selection_possible(ncu_obj)) {
1523                 nlog(LOG_DEBUG, "wlan_connect_thread: %s in invalid state or "
1524                     "has lower priority", ncu->ncu_name);
1525                 goto done;
1526         }

1528         /* If it is already connected to the required AP, just return. */
1529         if (nwamd_wlan_connected(ncu_obj)) {
1530                 nwamd_object_set_state(NWAM_OBJECT_TYPE_NCU,
1531                     ncu_obj->nwamd_object_name,
1532                     ncu_obj->nwamd_object_state, NWAM_AUX_STATE_UP);
1533                 goto done;
1534         }

1536         (void) memset(&attr, 0, sizeof (attr));
1537         if (dladm_wlan_str2essid(link->nwamd_link_wifi_essid, &attr.wa_essid)
1538             != DLADM_STATUS_OK) {
1539                 nlog(LOG_ERR, "wlan_connect_thread: invalid ESSID '%s' "
1540                     "for '%s'", link->nwamd_link_wifi_essid, ncu->ncu_name);
1541                 goto done;
1542         }
1543         attr.wa_valid = DLADM_WLAN_ATTR_ESSID;

1545         /* note: bssid logic here is non-functional */
1546         if (link->nwamd_link_wifi_bssid[0] != '\0') {
1547                 if (dladm_wlan_str2bssid(link->nwamd_link_wifi_bssid,
1548                     &attr.wa_bssid) != DLADM_STATUS_OK) {
1549                         nlog(LOG_ERR, "wlan_connect_thread: invalid BSSID '%s'",
1550                             "for '%s'", link->nwamd_link_wifi_bssid,
1551                             ncu->ncu_name);
1552                 } else {
1553                         attr.wa_valid |= DLADM_WLAN_ATTR_BSSID;
1554                 }
1555         }

1557         /* First check for the key */
1558         if (NEED_ENC(link->nwamd_link_wifi_security_mode)) {
1559                 if (link->nwamd_link_wifi_key == NULL) {
1560                         nlog(LOG_ERR, "wlan_connect_thread: could not find "
1561                             "key for WLAN '%s'", link->nwamd_link_wifi_essid);
1562                         nwamd_object_set_state(NWAM_OBJECT_TYPE_NCU,
1563                             ncu_obj->nwamd_object_name,
1564                             NWAM_STATE_OFFLINE_TO_ONLINE,
1565                             NWAM_AUX_STATE_LINK_WIFI_NEED_KEY);
1566                         goto done;
1567                 }
1568                 /* Make a copy of the key as we need to unlock the object */
1569                 if ((key = calloc(1, sizeof (dladm_wlan_key_t))) == NULL) {
1570                         nlog(LOG_ERR, "wlan_connect_thread: out of memory");
1571                         goto done;
1572                 }
1573                 (void) memcpy(key, link->nwamd_link_wifi_key,
1574                     sizeof (dladm_wlan_key_t));
```

```
1576                 attr.wa_valid |= DLADM_WLAN_ATTR_SECMODE;
1577                 attr.wa_secmode = link->nwamd_link_wifi_security_mode;
1578                 keycount = 1;
1579                 nlog(LOG_DEBUG, "wlan_connect_thread: retrieved key");
1580         } else {
1581                 key = NULL;
1582                 keycount = 0;
1583         }

1585         /*
1586          * Connect; only scan if a bssid was not specified.  If it times out,
1587          * try a second time using autoconf.  Drop the object lock during the
1588          * connect attempt since connecting may take some time, and access to
1589          * the link object during that period would be impossible if we held the
1590          * lock.
1591          */

1593         link->nwamd_link_wifi_autoconf = B_FALSE;
1594         link_id = link->nwamd_link_id;

1596         nwamd_object_release(ncu_obj);

1598         status = do_connect(link_id, &attr, key, keycount, 0);
 702         status = do_connect(link_id, &attr, key, keycount,
 703             DLADM_WLAN_CONNECT_NOSCAN);
1599         if (status != DLADM_STATUS_OK) {
1600                 /* Connect failed, try autoconf */
1601                 if (!wireless_autoconf || (status = do_connect(link_id, &attr,
1602                     NULL, 0, 0)) != DLADM_STATUS_OK) {
1603                         nlog(LOG_ERR, "wlan_connect_thread: connect failed for "
1604                             "%s", linkname);
1605                         goto done_unlocked;
1606                 }
1607                 if (status == DLADM_STATUS_OK)
1608                         autoconf = B_TRUE;
1609         }

1611         /* Connect succeeded, reacquire object */
1612         if ((ncu_obj = nwamd_ncu_object_find(NWAM_NCU_TYPE_LINK, linkname))
1613             == NULL) {
1614                 nlog(LOG_ERR, "wlan_connect_thread: could not find object  "
1615                     "for link %s", linkname);
1616                 goto done_unlocked;
1617         }

1619         ncu = ncu_obj->nwamd_object_data;
1620         link = &ncu->ncu_link;

1622         if (autoconf)
1623                 link->nwamd_link_wifi_autoconf = B_TRUE;

1625         /*
1626          * If WLAN is WEP/WPA, we would like to test the connection as the key
1627          * may be wrong.  It is difficult to find a reliable test that works
1628          * across APs however.  Do nothing for now.
1629          */
1630         link->nwamd_link_wifi_connected = nwamd_wlan_connected(ncu_obj);

1632         if (link->nwamd_link_wifi_connected) {
1633                 if (link->nwamd_link_wifi_add_to_known_wlans) {
1634                         /* add to known WLANs */
1635                         nlog(LOG_DEBUG, "wlan_connect_thread: "
1636                             "add '%s' to known WLANs",
1637                             link->nwamd_link_wifi_essid);
1638                         if ((err = nwam_known_wlan_add_to_known_wlans
1639                             (link->nwamd_link_wifi_essid,
```



```
1640                             link->nwamd_link_wifi_bssid[0] != '\0' ?
1641                             link->nwamd_link_wifi_bssid : NULL,
1642                             link->nwamd_link_wifi_security_mode,
1643                             link->nwamd_link_wifi_security_mode ==
1644                             DLADM_WLAN_SECMODE_WEP ?
1645                             (uint_t)link->nwamd_link_wifi_key->wk_idx : 1,
1646                             NEED_ENC(link->nwamd_link_wifi_security_mode) ?
1647                             link->nwamd_link_wifi_keyname : NULL))
1648                             != NWAM_SUCCESS) {
1649                                 nlog(LOG_ERR, "wlan_connect_thread: "
1650                                     "could not add to known WLANs: %s",
1651                                     nwam_strerror(err));
1652                         }
1653                 }
1654                 nwamd_set_selected_connected(ncu, B_TRUE, B_TRUE);
1655                 nlog(LOG_DEBUG, "wlan_connect_thread: connect "
1656                     "succeeded, setting state online");
1657                 nwamd_object_set_state(NWAM_OBJECT_TYPE_NCU,
1658                     ncu_obj->nwamd_object_name, NWAM_STATE_ONLINE,
1659                     NWAM_AUX_STATE_UP);
1660         }

1662 done:
1663         nwamd_object_release(ncu_obj);
1664 done_unlocked:
1665         free(linkname);
1666         free(key);

1668         return (NULL);
1669 }
```

unchanged_portion_omitted_

```
**********************************************************
    2056 Tue Jun 12 19:54:39 2012
new/usr/src/cmd/cmd-inet/usr.lib/Makefile
wpa_supplicant pkg now is created correctly in illumos-gate
wpad renamed to wpa_supplicant
**********************************************************
#
# 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) 1996, 2010, Oracle and/or its affiliates. All rights reserved.
#

SUBDIRS=        bridged dhcp dsvclockd ilbd in.chargend in.daytimed \
                in.discardd in.echod in.dhcpd in.mpathd in.ndpd \
                in.ripngd in.timed inetd mdnsd ncaconfd pppoe \
                slpd vrrpd wanboot wpa_supplicant
                slpd vrrpd wanboot wpad

MSGSUBDIRS=     dsvclockd ilbd in.dhcpd inetd ncaconfd vrrpd wanboot

include ../../Makefile.cmd
include ./Makefile.lib

$(CLOSED_BUILD)SUBDIRS += \
        $(CLOSED)/cmd/cmd-inet/usr.lib/ike-certutils \
        $(CLOSED)/cmd/cmd-inet/usr.lib/in.iked

POFILES=        dsvclockd/dsvclockd.po in.dhcpd/in.dhcpd.po \
                inetd/inetd.po ncaconfd/ncaconfd.po vrrpd/vrrpd.po \
                wanboot/wanboot.po
POFILE=         usr.lib.po

all:=           TARGET= all
install:=       TARGET= install
clean:=         TARGET= clean
clobber:=       TARGET= clobber
lint:=          TARGET= lint
_msg:=          TARGET= _msg

.KEEP_STATE:

all clean clobber lint: $(SUBDIRS)

install: $(SUBDIRS)
        -$(RM) $(ROOTLIBINET)/in.iked
        -$(LN) $(ISAEXEC) $(ROOTLIBINET)/in.iked
```



```
_msg: $(MSGSUBDIRS)

#
# The reason this rule checks for the existence of the
# Makefile is that some of the directories do not exist
# in our exportable source builds or in OpenSolaris.
#
$(SUBDIRS): FRC
        @if [ -f $@/Makefile  ]; then \
                cd $@; pwd; $(MAKE) $(TARGET); \
        else \
                true; \
        fi

FRC:
```

```
**********************************************************
    1337 Tue Jun 12 19:54:40 2012
new/usr/src/cmd/cmd-inet/usr.lib/wpa_supplicant/Makefile
wpa_supplicant pkg now is created correctly in illumos-gate
wpad renamed to wpa_supplicant
**********************************************************
```

```
#
# 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 2012 Sun Microsystems, Inc.  All rights reserved.
# Use is subject to license terms.
#

include ../../../Makefile.cmd

SUBDIR=         wpa_supplicant
MANIFEST =      wpa_supplicant.xml

ROOTMANIFESTDIR = $(ROOTSVCNETWORK)

all:=           TARGET= all
install:=       TARGET= install
clean:=         TARGET= clean
clobber:=       TARGET= clobber

.KEEP_STATE:

all lint clean clobber: $(SUBDIR)

include ../Makefile.lib

install:        $(SUBDIR) $(ROOTMANIFEST)

check:          $(CHKMANIFEST)

$(SUBDIR):      FRC
                @cd $@; pwd; $(MAKE) $(TARGET)

FRC:

include ../../../Makefile.targ
#endif /* ! codereview */
```

```
**********************************************************
    3472 Tue Jun 12 19:54:41 2012
new/usr/src/cmd/cmd-inet/usr.lib/wpa_supplicant/wpa_supplicant.xml
wpa_supplicant pkg now is created correctly in illumos-gate
wpad renamed to wpa_supplicant
**********************************************************
```

```
<?xml version="1.0"?>
<!DOCTYPE service_bundle SYSTEM "/usr/share/lib/xml/dtd/service_bundle.dtd.1">
<!--
        Copyright 2007 Sun Microsystems, Inc.  All rights reserved.
        Use is subject to license terms.

 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

        ident   "%Z%%M% %I%     %E% SMI"

        NOTE:  This service manifest is not editable; its contents will
        be overwritten by package or patch operations, including
        operating system upgrade.  Make customizations in a different
        file.
-->

<service_bundle type='manifest' name='SUNWsupr:wpa_supplicant'>

<service
        name='network/wpa_supplicant'
        type='service'
        version='1'>
        <single_instance/>
        <dependency
            name='cryptosvc'
            grouping='require_all'
            restart_on='none'
            type='service'>
                <service_fmri value='svc:/system/cryptosvc' />
        </dependency>

        <!--
            The wpa service will require the crypto framework for
            PKCS #11 keystores/certificates configuration
        -->

        <!--
            If wpa_supplicant crashes for some reasons,
            it will replace any existing file named
            /var/run/wpa_supplicant-global. So we do not need to manage the
            cleanup of the global UNIX socket.
            Socket clients are cleaned up in libdlwlan.
```



```
            B=background
            dd=debugging
            f=use syslog
            t=print timestamp
            W=wait for libdlwlan/dladm to add an interface before doing anything
        -->

        <exec_method
                type='method'
                name='start'
                exec='/usr/lib/inet/wpa_supplicant -B -f/var/log/wpa_supplicant.
                timeout_seconds='60'>
                        <method_context>
                                <method_credential
                                user='root'
                                group='root'
                                limit_privileges=':default'
                                privileges='basic,sys_net_config,net_rawaccess'
                                />
                        </method_context>
        </exec_method>

        <exec_method
                type='method'
                name='stop'
                exec=':kill'
                timeout_seconds='60'>
                        <method_context>
                                <method_credential
                                user='root'
                                group='root'
                                limit_privileges=':default'
                                privileges='basic,sys_net_config,net_rawaccess'
                                />
                        </method_context>
        </exec_method>

<!--
        <exec_method
                type='method'
                name='refresh'
                exec=':kill'
                timeout_seconds='60' />
-->

        <property_group name='general' type='framework'>
                <!-- to start stop wpad -->
                <propval name='action_authorization' type='astring'
                        value='solaris.smf.manage.wpa' />
        </property_group>

        <instance name="default" enabled="true">
        </instance>

        <stability value='External' />

        <template>
                <common_name>
                        <loctext xml:lang='C'>
                        Wireless WPA Supplicant
                        </loctext>
                </common_name>
                <documentation>
                        <manpage title='wpa_supplicant' section='1M'
                            manpath='/usr/share/man' />
                </documentation>
```

```
127         </template>
128 </service>

130 </service_bundle>
131 #endif /* ! codereview */
```

```
**********************************************************
    2983 Tue Jun 12 19:54:41 2012
new/usr/src/cmd/cmd-inet/usr.lib/wpa_supplicant/wpa_supplicant/Makefile
wpa_supplicant pkg now is created correctly in illumos-gate
wpad renamed to wpa_supplicant
**********************************************************
```

```
#
# 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 2012 Sun Microsystems, Inc.  All rights reserved.
# Use is subject to license terms.
#

PROG =          wpa_supplicant
OBJS =          config.o notify.o bss.o eap_register.o ../src/utils/common.o \
../src/utils/wpa_debug.o ../src/utils/wpabuf.o ../src/utils/os_unix.o \
../src/utils/eloop.o config_file.o ../src/rsn_supp/wpa.o \
../src/rsn_supp/preauth.o ../src/rsn_supp/pmksa_cache.o \
../src/rsn_supp/peerkey.o ../src/rsn_supp/wpa_ie.o ../src/common/wpa_common.o \
../src/eap_peer/eap_tls.o ../src/eap_peer/eap_peap.o \
../src/eap_common/eap_peap_common.o ../src/eap_peer/eap_ttls.o \
../src/eap_peer/eap_md5.o ../src/eap_peer/eap_mschapv2.o \
../src/eap_peer/mschapv2.o ../src/eapol_supp/eapol_supp_sm.o \
../src/eap_peer/eap.o ../src/eap_peer/eap_methods.o ../src/crypto/ms_funcs.o \
../src/eap_common/chap.o ../src/eap_peer/eap_tls_common.o \
../src/crypto/tls_openssl.o ../src/crypto/crypto_openssl.o \
../src/crypto/aes-unwrap.o ../src/crypto/md5.o ../src/crypto/random.o \
ctrl_iface.o ctrl_iface_unix.o ../src/utils/base64.o \
../src/eap_common/eap_common.o ../src/crypto/sha1.o ../src/crypto/sha1-pbkdf2.o
../src/crypto/sha1-tlsprf.o ../src/drivers/driver_common.o wpa_supplicant.o \
events.o blacklist.o wpas_glue.o scan.o main.o ../src/drivers/driver_solaris.o \
../src/drivers/drivers.o ../src/l2_packet/l2_packet_solaris.o

SRCS =          $(OBJS:%.o=%.c)

LINTME =        ../src/drivers/driver_solaris.c

include ../../../../Makefile.cmd

MSGFILES=       $(OBJS)

C99MODE =       $(C99_ENABLE)
CFLAGS +=       -I../src -I../src/utils
CFLAGS +=       -DCONFIG_BACKEND_FILE \
 -DCONFIG_DRIVER_SOLARIS -DEAP_TLS -DEAP_PEAP -DEAP_TTLS -DEAP_MD5 \
 -DEAP_MSCHAPv2 -DIEEE8021X_EAPOL -DPKCS12_FUNCS -DCONFIG_SMARTCARD \
 -DEAP_TLS_OPENSSL -DCONFIG_CTRL_IFACE -DCONFIG_CTRL_IFACE_UNIX \
 -DCONFIG_DEBUG_SYSLOG -DLOG_HOSTAPD="LOG_DAEMON" -DCONFIG_DEBUG_FILE
```



```
LDLIBS +=       -lsocket -ldlpi -ldladm

all install := LDLIBS += -lcrypto -lssl

LINTFLAGS +=    -u

.KEEP_STATE:

all:            $(PROG)

%.o:            %.c
                $(COMPILE.c) -o $@ $<

$(PROG):        $(OBJS)
                $(LINK.c) -o $@ $(OBJS) $(LDLIBS)
                $(POST_PROCESS)


include ../../Makefile.lib

install:        all $(ROOTLIBINETPROG)

clean:
                $(RM) $(OBJS)

include ../../../../Makefile.targ
#endif /* ! codereview */
```

```
**********************************************************
    8544 Tue Jun 12 19:54:41 2012
new/usr/src/cmd/cmd-inet/usr.sbin/Makefile
removed wificonfig tool
are /dev/wifi/* devices links now deprecated?
**********************************************************
#
# 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) 1990, 2010, Oracle and/or its affiliates. All rights reserved.
#

SYNCPROG=       syncinit syncloop syncstat
DHCPPROG=       dhcpconfig dhtadm pntadm

# EXPORT DELETE START
XMODPROG=       wanbootutil
# EXPORT DELETE END

PROG=           6to4relay arp gettable if_mpadm \
                in.comsat in.fingerd in.rarpd in.rexecd in.rlogind \
                in.rshd in.rwhod in.telnetd in.tftpd ipaddrsel \
                ndd $(SYNCPROG) $(DHCPPROG) $(XMODPROG)

MANIFEST=       rarp.xml telnet.xml comsat.xml finger.xml \
                login.xml shell.xml rexec.xml socket-filter-kssl.xml
SVCMETHOD=      svc-sockfilter

ROOTFS_PROG=    hostconfig route soconfig
SBINLINKS=      hostconfig route

RPCSVCPROG=     hostconfig
AUDITPROG=      in.rexecd in.rlogind in.rshd in.telnetd
PAMPROG=        in.rexecd in.rlogind in.rshd in.telnetd
SOCKETPROG=     6to4relay arp gettable hostconfig if_mpadm in.comsat \
                in.fingerd in.rarpd in.rexecd in.rlogind in.rshd \
                in.rwhod in.telnetd in.tftpd ipaddrsel route
NSLPROG=        6to4relay arp gettable hostconfig in.comsat in.rarpd \
                in.rexecd in.rlogind in.rshd in.rwhod in.telnetd \
                in.tftpd ipaddrsel route
CMDPROG=        in.telnetd
K5PROGS=        in.telnetd in.rlogind in.rshd
TSNETPROG=      route
DLADMPROG=      6to4relay
DEFAULTFILES=   telnetd.dfl

PROGSRCS=       $(PROG:%=%.c)
```



```
TFTPDOBJS=      in.tftpd.o tftpsubs.o
OTHERSRC=       ../usr.bin/tftp/tftpsubs.c
K5RLOGINOBJS=   in.rlogind.o
K5RSHDOBJS=     in.rshd.o
K5TELNETOBJS=   in.telnetd.o
SRCS=           $(PROGSRCS) $(OTHERSRC)

SUBDIRS=        bootconfchk htable ifconfig ilbadm in.ftpd in.rdisc in.routed \
                in.talkd inetadm inetconv ipadm ipmpstat ipqosconf ipsecutils \
                kssl/kssladm kssl/ksslcfg nwamadm nwamcfg ping routeadm \
                snoop sppptun traceroute
                snoop sppptun traceroute wificonfig

MSGSUBDIRS=     bootconfchk htable ifconfig ilbadm in.ftpd in.routed in.talkd \
                inetadm inetconv ipadm ipmpstat ipqosconf ipsecutils \
                kssl/ksslcfg nwamadm nwamcfg routeadm sppptun snoop
                kssl/ksslcfg nwamadm nwamcfg routeadm sppptun snoop wificonfig

# As programs get lint-clean, add them here and to the 'lint' target.
# Eventually this hack should go away, and all in PROG should be
# lint-clean.
LINTCLEAN=      6to4relay arp in.rlogind in.rshd in.telnetd in.tftpd \
                ipaddrsel route \
                in.rarpd if_mpadm $(SYNCPROG)
# Likewise, as subdirs get lint-clean, add them here.  Once
# they're all clean, replace the dependency of the lint target
# with SUBDIRS.  Also (sigh) deal with the commented-out build lines
# for the lint rule.
LINTSUBDIRS=    bootconfchk ilbadm in.rdisc in.routed in.talkd inetadm \
                inetconv ipmpstat ipqosconf ipsecutils kssl/kssladm \
                kssl/ksslcfg nwamadm nwamcfg ping routeadm sppptun traceroute
                kssl/ksslcfg nwamadm nwamcfg ping routeadm sppptun traceroute \
                wificonfig
# And as programs are verified not to attempt to write into constants,
# -xstrconst should be used to ensure they stay that way.
CONSTCLEAN=

include ../../Makefile.cmd
ROOTMANIFESTDIR=        $(ROOTSVCNETWORK)
$(ROOTMANIFEST)         := FILEMODE= 444
include ../Makefile.cmd-inet

ROOTSBINPROG = $(ROOTFS_PROG:%=$(ROOTSBIN)/%)
ROOTUSRSBINLINKS = $(SBINLINKS:%=$(ROOTUSRSBIN)/%)

COMMONOBJS=     addr_match.o kcmd.o store_forw_creds.o
COMMONSRCS=     $(COMMONOBJS:%.o=$(CMDINETCOMMONDIR)/%.c)
SRCS+=          $(COMMONSRCS)

#
# Message catalog
#
POFILES=        6to4relay.po if_mpadm.po in.comsat.po ipaddrsel.po route.po \
                soconfig.po
POFILE=         usr.sbin.po

all:=           TARGET= all
install:=       TARGET= install
clean:=         TARGET= clean
clobber:=       TARGET= clobber
lint:=          TARGET= lint
_msg:=          TARGET= _msg

CLOBBERFILES += $(ROOTFS_PROG) $(PROG)
CLEANFILES += $(COMMONOBJS) $(K5RLOGINOBJS) $(K5RSHDOBJS) $(TFTPDOBJS)
```

```
123 CPPFLAGS +=     -DSYSV -DBSD_COMP -I$(CMDINETCOMMONDIR) -I

125 include $(SRC)/lib/gss_mechs/mech_krb5/Makefile.mech_krb5
126 K5LIBS=

128 # Eventually just plain CFLAGS should be += -v, but not until all in
129 # PROGS are lint clean.
130 $(LINTCLEAN)            :=      CFLAGS += $(CCVERBOSE)
131 $(CONSTCLEAN)           :=      CFLAGS += $(XSTRCONST)

133 $(SYNCPROG)             :=      LDLIBS += -ldlpi
134 $(SOCKETPROG)           :=      LDLIBS += -lsocket
135 $(NSLPROG)              :=      LDLIBS += -lnsl
136 $(AUDITPROG)            :=      LDLIBS += -lbsm
137 $(PAMPROG)              :=      LDLIBS += -lpam
138 $(RPCSVCPROG)           :=      LDLIBS += -lrpcsvc
139 $(K5PROGS)              :=      LDFLAGS += $(KRUNPATH) \
140                                 -L$(ROOT)$(KLIBDIR_DO) -L$(ROOT)$(KLIBDIR_GL)
141 $(K5PROGS)              :=      K5LIBS=  -lmech_krb5
142 $(K5PROGS)              :=      CPPFLAGS += -I$(SRC)/head \
143                                 -I$(SRC)/uts/common/ \
144                                 -I$(SRC)/uts/common/gssapi/mechs/krb5/include \
145                                 -I$(SRC)/lib/gss_mechs/mech_krb5/include \
146                                 -I$(SRC)/lib/pam_modules/krb5
147 LDLIBS +=       $(K5LIBS)
148 $(TSNETPROG)            :=      LDLIBS += -ltsnet
149 $(DLADMPROG)            :=      LDLIBS += -ldladm

151 in.rarpd                :=      LDLIBS += -linetutil -ldlpi
152 if_mpadm                :=      LDLIBS += -linetutil -lipmp
153 if_mpadm.po             :=      XGETFLAGS += -a
154 route                   :=      CPPFLAGS += -DNDEBUG
155 ndd                     :=      LDLIBS += -ldladm -lipadm
156 gettable in.comsat      :=      LDFLAGS += $(MAPFILE.NGB:%=-M%)

158 .KEEP_STATE:

160 .PARALLEL:

162 all: $(PROG) $(ROOTFS_PROG) $(SUBDIRS) THIRDPARTYLICENSE.arp

164 #
165 # message catalog
166 #
167 _msg: $(MSGSUBDIRS) $(POFILE)

169 syncutil: $(SYNCPROG)

171 $(POFILE): $(POFILES)
172         $(RM) $@
173         cat $(POFILES) > $@

175 %.o: $(CMDINETCOMMONDIR)/%.c
176         $(COMPILE.c) -o $@ $<

178 in.telnetd: $(K5TELNETOBJS)
179         $(LINK.c) $(K5TELNETOBJS) -o $@ $(LDLIBS)
180         $(POST_PROCESS)

182 in.rlogind: $(K5RLOGINOBJS) $(COMMONOBJS)
183         $(LINK.c) $(K5RLOGINOBJS) $(COMMONOBJS) -o $@ $(LDLIBS)
184         $(POST_PROCESS)

186 in.rshd: $(K5RSHDOBJS) $(COMMONOBJS)
187         $(LINK.c) $(K5RSHDOBJS) $(COMMONOBJS) -o $@ $(LDLIBS)
188         $(POST_PROCESS)
```



```
190 in.tftpd: $(TFTPDOBJS)
191         $(LINK.c) $(TFTPDOBJS) -o $@ $(LDLIBS)
192         $(POST_PROCESS)

194 tftpsubs.o: $(OTHERSRC)
195         $(COMPILE.c) $(OTHERSRC) -o $@
196         $(POST_PROCESS_O)

198 $(ROOTUSRSBINLINKS):
199         -$(RM) $@; $(SYMLINK) ../../sbin/$(@F) $@

201 install: $(PROG) $(ROOTFS_PROG) $(SUBDIRS) .WAIT $(ROOTUSRSBINPROG) \
202         $(ROOTSBINPROG) $(ROOTUSRSBINLINKS) $(ROOTETCDEFAULTFILES) \
203         $(ROOTMANIFEST) $(ROOTSVCMETHOD) THIRDPARTYLICENSE.arp

205 THIRDPARTYLICENSE.arp: arp.c
206         $(SED) -n '/University of California/,/SUCH DAMAGE/p' arp.c > $@

208 CLOBBERFILES += THIRDPARTYLICENSE.arp

210 #
211 # The reason this rule checks for the existence of the
212 # Makefile is that some of the directories do not exist
213 # in our exportable source builds.
214 #
215 $(SUBDIRS): FRC
216         @if [ -f $@/Makefile  ]; then \
217                 cd $@; pwd; $(MAKE) $(TARGET); \
218         else \
219                 true; \
220         fi

222 FRC:

224 check:  $(CHKMANIFEST)

226 clean:  $(SUBDIRS)
227         -$(RM) $(CLEANFILES)

229 clobber: $(SUBDIRS)
230         -$(RM) $(CLEANFILES) $(CLOBBERFILES)

232 lint: $(LINTSUBDIRS)
233         $(LINT.c) 6to4relay.c $(LDLIBS) -lsocket -ldladm
234         $(LINT.c) arp.c $(LDLIBS) -lsocket -lnsl
235         @# $(LINT.c) in.rexecd.c $(LDLIBS) -lbsm -lpam
236         $(LINT.c) -erroff=E_NAME_USED_NOT_DEF2 -erroff=E_NAME_DEF_NOT_USED2 \
237                 -I$(SRC)/head -I$(SRC)/uts/common/ \
238                 -I$(SRC)/uts/common/gssapi/mechs/krb5/include \
239                 -I$(SRC)/lib/gss_mechs/mech_krb5/include \
240                 -I$(SRC)/lib/pam_modules/krb5 \
241                 in.rlogind.c $(COMMONSRCS) $(LDLIBS) -lbsm -lpam -lsocket -lnsl
242         $(LINT.c) -erroff=E_NAME_USED_NOT_DEF2 -erroff=E_NAME_DEF_NOT_USED2 \
243                 -I$(SRC)/head -I$(SRC)/uts/common/ \
244                 -I$(SRC)/uts/common/gssapi/mechs/krb5/include \
245                 -I$(SRC)/lib/gss_mechs/mech_krb5/include \
246                 -I$(SRC)/lib/pam_modules/krb5 \
247                 in.rshd.c $(COMMONSRCS) $(LDLIBS) -lbsm -lpam -lsocket -lnsl
248         $(LINT.c) -erroff=E_NAME_USED_NOT_DEF2 \
249                 -erroff=E_GLOBAL_COULD_BE_STATIC2 \
250                 -I$(SRC)/head -I$(SRC)/uts/common/ \
251                 -I$(SRC)/uts/common/gssapi/mechs/krb5/include \
252                 -I$(SRC)/lib/gss_mechs/mech_krb5/include \
253                 -I$(SRC)/lib/pam_modules/krb5 \
254                 in.telnetd.c $(LDLIBS) -lbsm -lpam -lsocket -lnsl
```

```
255         $(LINT.c) if_mpadm.c $(LDLIBS) -lsocket -lnsl -lipmp -linetutil
256         $(LINT.c) ipaddrsel.c $(LDLIBS) -lsocket -lnsl
257         $(LINT.c) route.c $(LDLIBS) -lsocket -lnsl -ltsnet
258         $(LINT.c) syncinit.c $(LDLIBS) -ldlpi
259         $(LINT.c) syncloop.c $(LDLIBS) -ldlpi
260         $(LINT.c) syncstat.c $(LDLIBS) -ldlpi
261         $(LINT.c) -erroff=E_NAME_USED_NOT_DEF2 in.rarpd.c $(LDLIBS) \
262             -lsocket -lnsl
263         $(LINT.c) in.tftpd.c ../usr.bin/tftp/tftpsubs.c $(LDLIBS) \
264             -lsocket -lnsl

266 # EXPORT DELETE START
267 EXPORT_SRC:
268         $(RM) Makefile+
269         sed -e "/^# EXPORT DELETE START/,/^# EXPORT DELETE END/d" \
270                 < Makefile > Makefile+
271         $(RM) Makefile
272         $(MV) Makefile+ Makefile
273         $(CHMOD) 444 Makefile
274 # EXPORT DELETE END
```

```
**********************************************************
  255013 Tue Jun 12 19:54:42 2012
new/usr/src/cmd/dladm/dladm.c
removed wpa.h header file
updated wpa_supplicant manifest to use global ctrl interface
some changes to libdlwlan, i need to finish ctrl_if integration
**********************************************************
______unchanged_portion_omitted_

 266 static cmd_t    cmds[] = {
 267         { "rename-link",        do_rename_link,
 268           "    rename-link      <oldlink> <newlink>"                 },
 269         { "show-link",          do_show_link,
 270           "    show-link        [-pP] [-o <field>,..] [-s [-i <interval>]] "
 271           "[<link>]\n"                                                },
 272         { "create-aggr",        do_create_aggr,
 273           "    create-aggr      [-t] [-P <policy>] [-L <mode>] [-T <time>] "
 274           "[-u <address>]\n"
 275           "\t\t     -l <link> [-l <link>...] <link>"                 },
 276         { "delete-aggr",        do_delete_aggr,
 277           "    delete-aggr      [-t] <link>"                         },
 278         { "add-aggr",           do_add_aggr,
 279           "    add-aggr         [-t] -l <link> [-l <link>...] <link>" },
 280         { "remove-aggr",        do_remove_aggr,
 281           "    remove-aggr      [-t] -l <link> [-l <link>...] <link>" },
 282         { "modify-aggr",        do_modify_aggr,
 283           "    modify-aggr      [-t] [-P <policy>] [-L <mode>] [-T <time>] "
 284           "[-u <address>]\n"
 285           "\t\t     <link>"                                          },
 286         { "show-aggr",          do_show_aggr,
 287           "    show-aggr        [-pPLx] [-o <field>,..] [-s [-i <interval>]] "
 288           "[<link>]\n"                                                },
 289         { "up-aggr",            do_up_aggr,     NULL                },
 290         { "scan-wifi",          do_scan_wifi,
 291           "    scan-wifi        [-p] [-o <field>,...] [<link>]"      },
 292         { "connect-wifi",       do_connect_wifi,
 293           "    connect-wifi     [-e <essid>] [-i <bssid>] [-k <key>,...]\n"
 294           "\t\t     [-s wep|wpa] [-u <username>]\n"
 293           "    connect-wifi     [-e <essid>] [-i <bssid>] [-k <key>,...] "
 294           "[-s wep|wpa]\n"
 295           "\t\t     [-a open|shared] [-b bss|ibss] [-c] [-m a|b|g] "
 296           "[-T <time>]\n"
 297           "\t\t     [<link>]"                                        },
 298         { "disconnect-wifi",    do_disconnect_wifi,
 299           "    disconnect-wifi  [-a] [<link>]"                       },
 300         { "show-wifi",          do_show_wifi,
 301           "    show-wifi        [-p] [-o <field>,...] [<link>]\n"    },
 302         { "set-linkprop",       do_set_linkprop,
 303           "    set-linkprop     [-t] -p <prop>=<value>[,...] <name>" },
 304         { "reset-linkprop",     do_reset_linkprop,
 305           "    reset-linkprop   [-t] [-p <prop>,...] <name>"         },
 306         { "show-linkprop",      do_show_linkprop,
 307           "    show-linkprop    [-cP] [-o <field>,...] [-p <prop>,...] "
 308           "<name>\n"                                                  },
 309         { "show-ether",         do_show_ether,
 310           "    show-ether       [-px][-o <field>,...] <link>\n"      },
 311         { "create-secobj",      do_create_secobj,
 312           "    create-secobj    [-t] [-f <file>] -c <class> <secobj>" },
 313         { "delete-secobj",      do_delete_secobj,
 314           "    delete-secobj    [-t] <secobj>[,...]"                 },
 315         { "show-secobj",        do_show_secobj,
 316           "    show-secobj      [-pP] [-o <field>,...] [<secobj>,...]\n" },
 317         { "init-linkprop",      do_init_linkprop,       NULL        },
 318         { "init-secobj",        do_init_secobj,         NULL        },
 319         { "create-vlan",        do_create_vlan,
 320           "    create-vlan      [-ft] -l <link> -v <vid> [link]"     },
```



```
 321         { "delete-vlan",        do_delete_vlan,
 322           "    delete-vlan      [-t] <link>"                         },
 323         { "show-vlan",          do_show_vlan,
 324           "    show-vlan        [-pP] [-o <field>,..] [<link>]\n"    },
 325         { "up-vlan",            do_up_vlan,             NULL        },
 326         { "create-iptun",       do_create_iptun,
 327           "    create-iptun     [-t] -T <type> "
 328           "[-a {local|remote}=<addr>,...] <link>]" },
 329         { "delete-iptun",       do_delete_iptun,
 330           "    delete-iptun     [-t] <link>"                         },
 331         { "modify-iptun",       do_modify_iptun,
 332           "    modify-iptun     [-t] -a {local|remote}=<addr>,... <link>" },
 333         { "show-iptun",         do_show_iptun,
 334           "    show-iptun       [-pP] [-o <field>,..] [<link>]\n"    },
 335         { "up-iptun",           do_up_iptun,            NULL        },
 336         { "down-iptun",         do_down_iptun,          NULL        },
 337         { "delete-phys",        do_delete_phys,
 338           "    delete-phys      <link>"                              },
 339         { "show-phys",          do_show_phys,
 340           "    show-phys        [-pP] [-o <field>,..] [-H] [<link>]\n"},
 341         { "init-phys",          do_init_phys,           NULL        },
 342         { "show-linkmap",       do_show_linkmap,        NULL        },
 343         { "create-vnic",        do_create_vnic,
 344           "    create-vnic      [-t] -l <link> [-m <value> | auto |\n"
 345           "\t\t     {factory [-n <slot-id>]} | {random [-r <prefix>]} |\n"
 346           "\t\t     {vrrp -V <vrid> -A {inet | inet6}} [-v <vid> [-f]]\n"
 347           "\t\t     [-p <prop>=<value>[,...]] <vnic-link>"    },
 348         { "delete-vnic",        do_delete_vnic,
 349           "    delete-vnic      [-t] <vnic-link>"                    },
 350         { "show-vnic",          do_show_vnic,
 351           "    show-vnic        [-pP] [-l <link>] [-s [-i <interval>]] "
 352           "[<link>]\n"                                                },
 353         { "up-vnic",            do_up_vnic,             NULL        },
 354         { "create-part",        do_create_part,
 355           "    create-part      [-t] [-f] -l <link> [-P <pkey>]\n"
 356           "\t\t     [-R <root-dir>] <part-link>"                     },
 357         { "delete-part",        do_delete_part,
 358           "    delete-part      [-t] [-R <root-dir>] <part-link>"},
 359         { "show-part",          do_show_part,
 360           "    show-part        [-pP] [-o <field>,...][-l <linkover>]\n"
 361           "\t\t     [<part-link>]"           },
 362         { "show-ib",            do_show_ib,
 363           "    show-ib          [-p] [-o <field>,...] [<link>]\n"    },
 364         { "up-part",            do_up_part,             NULL        },
 365         { "create-etherstub",   do_create_etherstub,
 366           "    create-etherstub [-t] <link>"                        },
 367         { "delete-etherstub",   do_delete_etherstub,
 368           "    delete-etherstub [-t] <link>"                        },
 369         { "show-etherstub",     do_show_etherstub,
 370           "    show-etherstub   [-t] [<link>]\n"                    },
 371         { "create-simnet",      do_create_simnet,       NULL        },
 372         { "modify-simnet",      do_modify_simnet,       NULL        },
 373         { "delete-simnet",      do_delete_simnet,       NULL        },
 374         { "show-simnet",        do_show_simnet,         NULL        },
 375         { "up-simnet",          do_up_simnet,           NULL        },
 376         { "create-bridge",      do_create_bridge,
 377           "    create-bridge    [-R <root-dir>] [-P <protect>] "
 378           "[-p <priority>]\n"
 379           "\t\t     [-m <max-age>] [-h <hello-time>] [-d <forward-delay>]\n"
 380           "\t\t     [-f <force-protocol>] [-l <link>]... <bridge>"    },
 381         { "modify-bridge",      do_modify_bridge,
 382           "    modify-bridge    [-R <root-dir>] [-P <protect>] "
 383           "[-p <priority>]\n"
 384           "\t\t     [-m <max-age>] [-h <hello-time>] [-d <forward-delay>]\n"
 385           "\t\t     [-f <force-protocol>] <bridge>"                  },
 386         { "delete-bridge",      do_delete_bridge,
```

```
 387             "    delete-bridge    [-R <root-dir>] <bridge>"             },
 388         { "add-bridge",          do_add_bridge,
 389             "    add-bridge       [-R <root-dir>] -l <link> [-l <link>]... "
 390             "<bridge>"                                                  },
 391         { "remove-bridge",       do_remove_bridge,
 392             "    remove-bridge    [-R <root-dir>] -l <link> [-l <link>]... "
 393             "<bridge>"                                                  },
 394         { "show-bridge",         do_show_bridge,
 395             "    show-bridge      [-p] [-o <field>,...] [-s [-i <interval>]] "
 396             "[<bridge>]\n"
 397             "    show-bridge      -l [-p] [-o <field>,...] [-s [-i <interval>]]"
 398             " <bridge>\n"
 399             "    show-bridge      -f [-p] [-o <field>,...] [-s [-i <interval>]]"
 400             " <bridge>\n"
 401             "    show-bridge      -t [-p] [-o <field>,...] [-s [-i <interval>]]"
 402             " <bridge>\n"                                               },
 403         { "show-usage",          do_show_usage,
 404             "    show-usage       [-a] [-d | -F <format>] "
 405             "[-s <DD/MM/YYYY,HH:MM:SS>]\n"
 406             "\t\t     [-e <DD/MM/YYYY,HH:MM:SS>] -f <logfile> [<link>]" }
 407 };
_____unchanged_portion_omitted_

6254 static void
6255 do_connect_wifi(int argc, char **argv, const char *use)
6256 {
6257         int                     option;
6258         dladm_wlan_attr_t       attr, *attrp;
6259         dladm_status_t          status = DLADM_STATUS_OK;
6260         int                     timeout = DLADM_WLAN_CONNECT_TIMEOUT_DEFAULT;
6261         datalink_id_t           linkid = DATALINK_ALL_LINKID;
6262         dladm_wlan_key_t        *keys = NULL;
6263         uint_t                  key_count = 0;
6264         uint_t                  flags = 0;
6265         dladm_wlan_secmode_t    keysecmode = DLADM_WLAN_SECMODE_NONE;
6266         char                    buf[DLADM_STRSIZE];
6267         char                    identity[128];
6268 #endif /* ! codereview */

6270         opterr = 0;
6271         (void) memset(&attr, 0, sizeof (attr));
6272         while ((option = getopt_long(argc, argv, ":e:i:a:m:b:s:k:u:T:c",
6267         while ((option = getopt_long(argc, argv, ":e:i:a:m:b:s:k:T:c",
6273             wifi_longopts, NULL)) != -1) {
6274                 switch (option) {
6275                 case 'e':
6276                         status = dladm_wlan_str2essid(optarg, &attr.wa_essid);
6277                         if (status != DLADM_STATUS_OK)
6278                                 die("invalid ESSID '%s'", optarg);

6280                         attr.wa_valid |= DLADM_WLAN_ATTR_ESSID;
6276                         /*
6277                          * Try to connect without doing a scan.
6278                          */
6279                         flags |= DLADM_WLAN_CONNECT_NOSCAN;
6281                         break;
6282                 case 'i':
6283                         status = dladm_wlan_str2bssid(optarg, &attr.wa_bssid);
6284                         if (status != DLADM_STATUS_OK)
6285                                 die("invalid BSSID %s", optarg);

6287                         attr.wa_valid |= DLADM_WLAN_ATTR_BSSID;
6288                         break;
6289                 case 'a':
6290                         status = dladm_wlan_str2auth(optarg, &attr.wa_auth);
6291                         if (status != DLADM_STATUS_OK)
```



```
6292                                 die("invalid authentication mode '%s'", optarg);

6294                         attr.wa_valid |= DLADM_WLAN_ATTR_AUTH;
6295                         break;
6296                 case 'm':
6297                         status = dladm_wlan_str2mode(optarg, &attr.wa_mode);
6298                         if (status != DLADM_STATUS_OK)
6299                                 die("invalid mode '%s'", optarg);

6301                         attr.wa_valid |= DLADM_WLAN_ATTR_MODE;
6302                         break;
6303                 case 'b':
6304                         if ((status = dladm_wlan_str2bsstype(optarg,
6305                             &attr.wa_bsstype)) != DLADM_STATUS_OK) {
6306                                 die("invalid bsstype '%s'", optarg);
6307                         }

6309                         attr.wa_valid |= DLADM_WLAN_ATTR_BSSTYPE;
6310                         break;
6311                 case 's':
6312                         if ((status = dladm_wlan_str2secmode(optarg,
6313                             &attr.wa_secmode)) != DLADM_STATUS_OK) {
6314                                 die("invalid security mode '%s'", optarg);
6315                         }

6317                         attr.wa_valid |= DLADM_WLAN_ATTR_SECMODE;
6318                         break;
6319                 case 'k':
6320                         if (parse_wlan_keys(optarg, &keys, &key_count) < 0)
6321                                 die("invalid key(s) '%s'", optarg);

6323                         if (keys[0].wk_class == DLADM_SECOBJ_CLASS_WEP)
6324                                 keysecmode = DLADM_WLAN_SECMODE_WEP;
6325                         else if (keys[0].wk_class == DLADM_SECOBJ_CLASS_PSK)
6326                                 keysecmode = DLADM_WLAN_SECMODE_PSK;
6327 #endif /* ! codereview */
6328                         else
6329                                 keysecmode = DLADM_WLAN_SECMODE_EAP;
6330                         break;
6331                 case 'u':
6332                         if (snprintf(identity, 131, "\"%s\"", optarg) <= 0)
6333                                 die("invalid username", optarg);
6324                         keysecmode = DLADM_WLAN_SECMODE_WPA;
6334                         break;
6335                 case 'T':
6336                         if (strcasecmp(optarg, "forever") == 0) {
6337                                 timeout = -1;
6338                                 break;
6339                         }
6340                         if (!str2int(optarg, &timeout) || timeout < 0)
6341                                 die("invalid timeout value '%s'", optarg);
6342                         break;
6343                 case 'c':
6344                         flags |= DLADM_WLAN_CONNECT_CREATEIBSS;
6345                         flags |= DLADM_WLAN_CONNECT_CREATEIBSS;
6346                         break;
6347                 default:
6348                         die_opterr(optopt, option, use);
6349                         break;
6350                 }
6351         }

6353         if (keysecmode == DLADM_WLAN_SECMODE_NONE) {
6354                 if ((attr.wa_valid & DLADM_WLAN_ATTR_SECMODE) != 0)
6345                 if ((attr.wa_valid & DLADM_WLAN_ATTR_SECMODE) != 0) {
6355                         die("key required for security mode '%s'",
```

```
6356                            dladm_wlan_secmode2str(&attr.wa_secmode, buf));
6348                    }
6357            } else {
6358                    if ((attr.wa_valid & DLADM_WLAN_ATTR_SECMODE) != 0 &&
6359                        attr.wa_secmode != keysecmode)
6360                            die("incompatible -s and -k options");
6361                    if ((keysecmode == DLADM_WLAN_SECMODE_EAP) != 0 &&
6362                        identity == NULL)
6363                            die("username (-u) required for security mode '%s'",
6364                                dladm_wlan_secmode2str(&attr.wa_secmode, buf));
6365 #endif /* ! codereview */
6366                    attr.wa_valid |= DLADM_WLAN_ATTR_SECMODE;
6367                    attr.wa_secmode = keysecmode;
6368            }

6370            if (optind == (argc - 1)) {
6371                    if ((status = dladm_name2info(handle, argv[optind], &linkid,
6372                        NULL, NULL, NULL)) != DLADM_STATUS_OK) {
6373                            die_dlerr(status, "link %s is not valid", argv[optind]);
6374                    }
6375            } else if (optind != argc) {
6376                    usage();
6377            }

6379            if (linkid == DATALINK_ALL_LINKID) {
6380                    wlan_count_attr_t wcattr;

6382                    wcattr.wc_linkid = DATALINK_INVALID_LINKID;
6383                    wcattr.wc_count = 0;
6384                    (void) dladm_walk_datalink_id(do_count_wlan, handle, &wcattr,
6385                        DATALINK_CLASS_PHYS | DATALINK_CLASS_SIMNET,
6386                        DL_WIFI, DLADM_OPT_ACTIVE);
6387                    if (wcattr.wc_count == 0) {
6388                            die("no wifi links are available");
6389                    } else if (wcattr.wc_count > 1) {
6390                            die("link name is required when more than one wifi "
6391                                "link is available");
6392                    }
6393                    linkid = wcattr.wc_linkid;
6394            }
6395            attrp = (attr.wa_valid == 0) ? NULL : &attr;

6353 again:
6397            if ((status = dladm_wlan_connect(handle, linkid, attrp, timeout, keys,
6398                key_count, flags, identity)) != DLADM_STATUS_OK) {
6355                key_count, flags)) != DLADM_STATUS_OK) {
6356                    if ((flags & DLADM_WLAN_CONNECT_NOSCAN) != 0) {
6357                            /*
6358                             * Try again with scanning and filtering.
6359                             */
6360                            flags &= ~DLADM_WLAN_CONNECT_NOSCAN;
6361                            goto again;
6362                    }

6399                    if (status == DLADM_STATUS_NOTFOUND) {
6400                            if (attr.wa_valid == 0) {
6401                                    die("no wifi networks are available");
6402                            } else {
6403                                    die("no wifi networks with the specified "
6404                                        "criteria are available");
6405                            }
6406                    }
6407                    die_dlerr(status, "cannot connect");
6408            }
6409            free(keys);
6410 }
```

_____unchanged_portion_omitted_


```
7039 static int
7040 convert_secobj(char *buf, uint_t len, uint8_t *obj_val, uint_t *obj_lenp,
7041     dladm_secobj_class_t class)
7042 {
7043            int error = 0;

7045            if (class != DLADM_SECOBJ_CLASS_WEP) {
7046                    if ((class == DLADM_SECOBJ_CLASS_PSK) && (len < 8 || len > 63))
7010            if (class == DLADM_SECOBJ_CLASS_WPA) {
7011                    if (len < 8 || len > 63)
7047                            return (EINVAL);
7048                    (void) memcpy(obj_val, buf, len);
7049                    *obj_lenp = len;
7050                    return (error);
7051            } else {
7016            }

7018            if (class == DLADM_SECOBJ_CLASS_WEP) {
7052                    switch (len) {
7053                    case 5:                 /* ASCII key sizes */
7054                    case 13:
7055                            (void) memcpy(obj_val, buf, len);
7056                            *obj_lenp = len;
7057                            break;
7058                    case 10:                /* Hex key sizes, not preceded by 0x */
7059                    case 26:
7060                            error = hexascii_to_octet(buf, len, obj_val, obj_lenp);
7061                            break;
7062                    case 12:                /* Hex key sizes, preceded by 0x */
7063                    case 28:
7064                            if (strncmp(buf, "0x", 2) != 0)
7065                                    return (EINVAL);
7066                            error = hexascii_to_octet(buf + 2, len - 2,
7067                                obj_val, obj_lenp);
7068                            break;
7069                    default:
7070                            return (EINVAL);
7071                    }
7072                    return (error);
7073            }

7075            return (ENOENT);
7076 }
```

_____unchanged_portion_omitted_

```
**********************************************************
   15413 Tue Jun 12 19:54:49 2012
new/usr/src/cmd/svc/milestone/net-physical
removed wificonfig tool
are /dev/wifi/* devices links now deprecated?
**********************************************************
______unchanged_portion_omitted_

#
# All the IPv4 and IPv6 interfaces are plumbed before doing any
# interface configuration.  This prevents errors from plumb failures
# getting mixed in with the configured interface lists that the script
# outputs.
#

#
# First deal with /etc/hostname
#
# Get the list of IPv4 interfaces to configure by breaking
# /etc/hostname.* into separate args by using "." as a shell separator
# character.
#
interface_names="`echo /etc/hostname.*[0-9] 2>/dev/null`"
if [ "$interface_names" != "/etc/hostname.*[0-9]" ]; then
        ORIGIFS="$IFS"
        IFS="$IFS."
        set -- $interface_names
        IFS="$ORIGIFS"
        while [ $# -ge 2 ]; do
                shift
                intf_name=$1
                while [ $# -gt 1 -a "$2" != "/etc/hostname" ]; do
                        intf_name="$intf_name.$2"
                        shift
                done
                shift

                # skip IP tunnel interfaces plumbed by net-iptun.
                if is_iptun $intf_name; then
                        continue
                fi

                read one rest < /etc/hostname.$intf_name
                if [ "$one" = ipmp ]; then
                        ipmp_list="$ipmp_list $intf_name"
                else
                        inet_list="$inet_list $intf_name"
                fi
        done
fi

#
# Get the list of IPv6 interfaces to configure by breaking
# /etc/hostname6.* into separate args by using "." as a shell separator
# character.
#
interface_names="`echo /etc/hostname6.*[0-9] 2>/dev/null`"
if [ "$interface_names" != "/etc/hostname6.*[0-9]" ]; then
        ORIGIFS="$IFS"
        IFS="$IFS."
        set -- $interface_names
        IFS="$ORIGIFS"
        while [ $# -ge 2 ]; do
                shift
                intf_name=$1
                while [ $# -gt 1 -a "$2" != "/etc/hostname6" ]; do
```



```
                        intf_name="$intf_name.$2"
                        shift
                done
                shift

                # skip IP tunnel interfaces plumbed by net-iptun.
                if is_iptun $intf_name; then
                        continue
                fi

                read one rest < /etc/hostname6.$intf_name
                if [ "$one" = ipmp ]; then
                        ipmp6_list="$ipmp6_list $intf_name"
                else
                        inet6_list="$inet6_list $intf_name"
                fi
        done
fi

#
# Create all of the IPv4 IPMP interfaces.
#
if [ -n "$ipmp_list" ]; then
        set -- $ipmp_list
        while [ $# -gt 0 ]; do
                if /sbin/ifconfig $1 ipmp; then
                        ipmp_created="$ipmp_created $1"
                else
                        ipmp_failed="$ipmp_failed $1"
                fi
                shift
        done
        [ -n "$ipmp_failed" ] && warn_failed_ifs "create IPv4 IPMP" \
            "$ipmp_failed"
fi

#
# Step through the IPv4 interface list and try to plumb every interface.
# Generate list of plumbed and failed IPv4 interfaces.
#
if [ -n "$inet_list" ]; then
        set -- $inet_list
        while [ $# -gt 0 ]; do
                /sbin/ifconfig $1 plumb
                if /sbin/ifconfig $1 inet >/dev/null 2>&1; then
                        inet_plumbed="$inet_plumbed $1"
                else
                        inet_failed="$inet_failed $1"
                fi
                shift
        done
        [ -n "$inet_failed" ] && warn_failed_ifs "plumb IPv4" "$inet_failed"
fi

# Run autoconf to connect to a WLAN if the interface is a wireless one
if [ -x /sbin/wificonfig -a -n "$inet_plumbed" ]; then
        set -- $inet_plumbed
        while [ $# -gt 0 ]; do
                        if [ -r /dev/wifi/$1 ]; then
                                /sbin/wificonfig -i $1 startconf >/dev/null
                        fi
                shift
        done
fi

#
```

```
263 # Step through the IPv6 interface list and plumb every interface.
264 # Generate list of plumbed and failed IPv6 interfaces.  Each plumbed
265 # interface will be brought up later, after processing any contents of
266 # the /etc/hostname6.* file.
267 #
268 if [ -n "$inet6_list" ]; then
269         set -- $inet6_list
270         while [ $# -gt 0 ]; do
271                 /sbin/ifconfig $1 inet6 plumb
272                 if /sbin/ifconfig $1 inet6 >/dev/null 2>&1; then
273                         inet6_plumbed="$inet6_plumbed $1"
274                 else
275                         inet6_failed="$inet6_failed $1"
276                 fi
277                 shift
278         done
279         [ -n "$inet6_failed" ] && warn_failed_ifs "plumb IPv6" "$inet6_failed"
280 fi

282 #
283 # Create all of the IPv6 IPMP interfaces.
284 #
285 if [ -n "$ipmp6_list" ]; then
286         set -- $ipmp6_list
287         while [ $# -gt 0 ]; do
288                 if /sbin/ifconfig $1 inet6 ipmp; then
289                         ipmp6_created="$ipmp6_created $1"
290                 else
291                         ipmp6_failed="$ipmp6_failed $1"
292                 fi
293                 shift
294         done
295         [ -n "$ipmp6_failed" ] && warn_failed_ifs "create IPv6 IPMP" \
296             "$ipmp6_failed"
297 fi

299 #
300 # Finally configure interfaces set up with ipadm. Any /etc/hostname*.intf
301 # files take precedence over ipadm defined configurations except when
302 # we are in a non-global zone and Layer-3 protection of IP addresses is
303 # enforced on the interface by the global zone.
304 #
305 for showif_output in `/sbin/ipadm show-if -p -o ifname,state,current`; do
306         intf=`echo $showif_output | /usr/bin/cut -f1 -d:`
307         state=`echo $showif_output | /usr/bin/cut -f2 -d:`
308         current=`echo $showif_output | /usr/bin/cut -f3 -d:`
309         if  [[ "$state" != "disabled" && $current != *Z* ]]; then
310                 #
311                 # skip if not a persistent interface, or if it should get IP
312                 # configuration from the global zone ('Z' flag is set)
313                 #
314                 continue;
315         elif is_iptun $intf; then
316                 # skip IP tunnel interfaces plumbed by net-iptun
317                 continue;
318         elif [ -f /etc/hostname.$intf ] || [ -f /etc/hostname6.$intf ]; then
319                 if [[ $current != *Z* ]]; then
320                         echo "found /etc/hostname.$intf "\
321                             "or /etc/hostname6.$intf, "\
322                             "ignoring ipadm configuration" > /dev/msglog
323                         continue;
324                 else
325                         echo "Ignoring /etc/hostname*.$intf" > /dev/msglog
326                         /sbin/ifconfig $intf unplumb > /dev/null 2>&1
327                         /sbin/ifconfig $intf inet6 unplumb > /dev/null 2>&1
328                 fi
```



```
329         fi

331         # Enable the interface managed by ipadm
332         /sbin/ipadm enable-if -t $intf
333 done

335 #
336 # Process the /etc/hostname[6].* files for IPMP interfaces.  Processing these
337 # before non-IPMP interfaces avoids accidental implicit IPMP group creation.
338 #
339 [ -n "$ipmp_created" ] && if_configure inet "IPMP" $ipmp_created
340 [ -n "$ipmp6_created" ] && if_configure inet6 "IPMP" $ipmp6_created

342 #
343 # Process the /etc/hostname[6].* files for non-IPMP interfaces.
344 #
345 [ -n "$inet_plumbed" ] && if_configure inet "" $inet_plumbed
346 [ -n "$inet6_plumbed" ] && if_configure inet6 "" $inet6_plumbed

348 #
349 # For the IPv4 and IPv6 interfaces that failed to plumb, find (or create)
350 # IPMP meta-interfaces to host their data addresses.
351 #
352 [ -n "$inet_failed" ] && move_addresses inet
353 [ -n "$inet6_failed" ] && move_addresses inet6

355 # Run DHCP if requested. Skip boot-configured interface.
356 interface_names="`echo /etc/dhcp.*[0-9] 2>/dev/null`"
357 if [ "$interface_names" != '/etc/dhcp.*[0-9]' ]; then
358         #
359         # First find the primary interface. Default to the first
360         # interface if not specified. First primary interface found
361         # "wins". Use care not to "reconfigure" a net-booted interface
362         # configured using DHCP. Run through the list of interfaces
363         # again, this time trying DHCP.
364         #
365         i4d_fail=
366         firstif=
367         primary=
368         ORIGIFS="$IFS"
369         IFS="${IFS}."
370         set -- $interface_names

372         while [ $# -ge 2 ]; do
373                 shift
374                 [ -z "$firstif" ] && firstif=$1

376                 for i in `shcat /etc/dhcp\.$1`; do
377                         if [ "$i" = primary ]; then
378                                 primary=$1
379                                 break
380                         fi
381                 done

383                 [ -n "$primary" ] && break
384                 shift
385         done

387         [ -z "$primary" ] && primary="$firstif"
388         cmdline=`shcat /etc/dhcp\.${primary}`

390         if [ "$_INIT_NET_IF" != "$primary" ]; then
391                 echo "starting DHCP on primary interface $primary"
392                 /sbin/ifconfig $primary auto-dhcp primary $cmdline
393                 # Exit code 4 means ifconfig timed out waiting for dhcpagent
394                 [ $? != 0 ] && [ $? != 4 ] && i4d_fail="$i4d_fail $primary"
```

```
395         fi
397         set -- $interface_names
399         while [ $# -ge 2 ]; do
400                 shift
401                 cmdline=‘shcat /etc/dhcp\.$1‘
402                 if [ "$1" != "$primary" -a \
403                         "$1" != "$_INIT_NET_IF"  ]; then
404                         echo "starting DHCP on interface $1"
405                         /sbin/ifconfig $1 dhcp start wait 0 $cmdline
406                         # Exit code can’t be timeout when wait is 0
407                         [ $? != 0 ] && i4d_fail="$i4d_fail $1"
408                 fi
409                 shift
410         done
411         IFS="$ORIGIFS"
412         unset ORIGIFS
413         [ -n "$i4d_fail" ] && warn_failed_ifs "configure IPv4 DHCP" "$i4d_fail"
414 fi
416 # In order to avoid bringing up the interfaces that have
417 # intentionally been left down, perform RARP only if the system
418 # has no configured hostname in /etc/nodename
419 hostname="‘shcat /etc/nodename 2>/dev/null‘"
420 if [ "$_INIT_NET_STRATEGY" = "rarp" -o -z "$hostname" ]; then
421         /sbin/ifconfig -adD4 auto-revarp netmask + broadcast + up
422 fi
424 #
425 # If the /etc/defaultrouter file exists, process it now so that the next
426 # stage of booting will have access to NFS.
427 #
428 if [ -f /etc/defaultrouter ]; then
429         while read router rubbish; do
430                 case "$router" in
431                         ‘#‘* | ‘‘) ;;   #  Ignore comments, empty lines
432                         *)      /sbin/route -n add default -gateway $router ;;
433                 esac
434         done </etc/defaultrouter
435 fi
437 #
438 # If we get here and were not asked to plumb any IPv4 interfaces, look
439 # for boot properties that direct us.
440 #
441 # - The "network-interface" property is required and indicates the
442 #   interface name.
443 # - The "xpv-hcp" property, if present, is used by the hypervisor
444 #   tools to indicate how the specified interface should be configured.
445 #   Permitted values are "dhcp" and "off", where "off" indicates static
446 #   IP configuration.
447 #
448 # In the case where "xpv-hcp" is set to "dhcp", no further properties
449 # are required or examined.
450 #
451 # In the case where "xpv-hcp" is not present or set to "off", the
452 # "host-ip" and "subnet-mask" properties are used to configure
453 # the specified interface.  The "router-ip" property, if present,
454 # is used to add a default route.
455 #
456 nic="‘/sbin/devprop network-interface‘"
457 if smf_is_globalzone && [ -z "$inet_list" ] && [ -n "$nic" ]; then
458         hcp="‘/sbin/devprop xpv-hcp‘"
459         case "$hcp" in
460         "dhcp")
```



```
461                 /sbin/ifconfig $nic plumb 2>/dev/null
462                 [ -n "‘/sbin/ifconfig $nic 2>/dev/null‘" ] && (
463                         # The interface is successfully plumbed, so
464                         # modify "inet_list" to force the exit code
465                         # checks to work.
466                         inet_list=$nic;
467                         # Given that this is the only IPv4 interface,
468                         # we assert that it is primary.
469                         echo "starting DHCP on primary interface $primary";
470                         /sbin/ifconfig $nic auto-dhcp primary;
471                         # Exit code 4 means ifconfig timed out waiting
472                         # for dhcpagent
473                         [ $? != 0 ] && [ $? != 4 ] && \
474                             i4d_fail="$i4d_fail $nic";
475                 )
476                 ;;
478         "off"|"")
479                 /sbin/devprop host-ip subnet-mask router-ip | (
480                         read ip;
481                         read mask;
482                         read router;
483                         [ -n "$ip" ] && [ -n "$mask" ] && \
484                                 /sbin/ifconfig $nic plumb 2>/dev/null
485                         [ -n "‘/sbin/ifconfig $nic 2>/dev/null‘" ] && (
486                                 # The interface is successfully
487                                 # plumbed, so modify "inet_list" to
488                                 # force the exit code checks to work.
489                                 inet_list=$nic;
490                                 /sbin/ifconfig $nic  inet $ip \
491                                     netmask $mask broadcast + up 2>/dev/null;
492                                 [ -n "$router" ] && route add \
493                                     default $router 2>/dev/null;
494                         )
495                 )
496                 ;;
497         esac
498 fi
500 #
501 # We tell smf this service is online if any of the following is true:
502 # - no interfaces were configured for plumbing and no DHCP failures
503 # - any non-loopback IPv4 interfaces are up and have a non-zero address
504 # - there are any DHCP interfaces started
505 # - any non-loopback IPv6 interfaces are up
506 #
507 # If we weren’t asked to configure any interfaces, exit
508 if [ -z "$inet_list" ] && [ -z "$inet6_list" ]; then
509         # Config error if DHCP was attempted without plumbed interfaces
510         [ -n "$i4d_fail" ] && exit $SMF_EXIT_ERR_CONFIG
511         exit $SMF_EXIT_OK
512 fi
514 # Any non-loopback IPv4 interfaces with usable addresses up?
515 if [ -n "‘/sbin/ifconfig -a4u‘" ]; then
516         /sbin/ifconfig -a4u | while read intf addr rest; do
517                 [ $intf = inet ] && [ $addr != 127.0.0.1 ] &&
518                 [ $addr != 0.0.0.0 ] && exit $SMF_EXIT_OK
519         done && exit $SMF_EXIT_OK
520 fi
522 # Any DHCP interfaces started?
523 [ -n "‘/sbin/ifconfig -a4 dhcp status 2>/dev/null‘" ] && exit $SMF_EXIT_OK
525 # Any non-loopback IPv6 interfaces up?
526 if [ -n "‘/sbin/ifconfig -au6‘" ]; then
```

```
 527         /sbin/ifconfig -au6 | while read intf addr rest; do
 528                 [ $intf = inet6 ] && [ $addr != ::1/128 ] && exit $SMF_EXIT_OK
 529         done && exit $SMF_EXIT_OK
 530 fi

 532 # This service was supposed to configure something yet didn't.  Exit
 533 # with config error.
 534 exit $SMF_EXIT_ERR_CONFIG
```

```
**********************************************************
    5126 Tue Jun 12 19:54:50 2012
new/usr/src/cmd/svc/milestone/network-physical.xml
removed wpa.h header file
updated wpa_supplicant manifest to use global ctrl interface
some changes to libdlwlan, i need to finish ctrl_if integration
**********************************************************
```

```
<?xml version="1.0"?>
<!DOCTYPE service_bundle SYSTEM "/usr/share/lib/xml/dtd/service_bundle.dtd.1">
<!--
 Copyright 2010 Sun Microsystems, Inc.  All rights reserved.
 Use is subject to license terms.

 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

        NOTE:  This service manifest is not editable; its contents will
        be overwritten by package or patch operations, including
        operating system upgrade.  Make customizations in a different
        file.
-->

<service_bundle type='manifest' name='SUNWcsr:network-physical'>

<service
        name='network/physical'
        type='service'
        version='1'>

        <!-- ifconfig needs loopback for IPC with dhcpagent -->
        <dependency
                name='loopback'
                grouping='require_all'
                restart_on='none'
                type='service'>
                <service_fmri value='svc:/network/loopback' />
        </dependency>

        <instance name='default' enabled='true'>

        <!--
                physical:default and physical:nwam are mutually exclusive.
                Use a one-way dependency for now since two-way exclude_all
                does not work; enforcement of single_instance in the future
                will fix this.
        -->
        <dependency
                name='physical_nwam'
                grouping='exclude_all'
                restart_on='none'
```



```
                type='service'>
                <service_fmri value='svc:/network/physical:nwam' />
        </dependency>

        <exec_method
                type='method'
                name='start'
                exec='/lib/svc/method/net-physical'
                timeout_seconds='600' />

        <exec_method
                type='method'
                name='stop'
                exec=':true'
                timeout_seconds='3' />

        <property_group name='startd' type='framework'>
                <propval name='duration' type='astring' value='transient' />
        </property_group>

        <template>
                <common_name>
                        <loctext xml:lang='C'>
                                physical network interfaces
                        </loctext>
                </common_name>
                <documentation>
                        <manpage title='ifconfig' section='1M'
                                manpath='/usr/share/man' />
                </documentation>
        </template>

        </instance>

        <instance name='nwam' enabled='false'>

        <!--
                wpa_supplicant just opens global control interface and waits for
                wpa_ctrl to add an interface (could be wired OR wireless)
        -->
        <!--
        <dependency
                name='wpa_supplicant'
                grouping='require_all'
                restart_on='none'
                type='service'>
                <service_fmri value='svc:/network/wpa_supplicant' />
        </dependency>
        -->

#endif /* ! codereview */
        <exec_method
                type='method'
                name='start'
                exec='/lib/svc/method/net-nwam start'
                timeout_seconds='120' >
                <method_context>
                        <method_credential user='root' group='root'
                                supp_groups='netadm' privileges='zone' />
                </method_context>
        </exec_method>

        <exec_method
                type='method'
                name='stop'
                exec='/lib/svc/method/net-nwam stop'
```

```
126                 timeout_seconds='60' >
127                 <method_context>
128                         <method_credential user='root' group='root'
129                                 supp_groups='netadm' privileges='zone' />
130                 </method_context>
131         </exec_method>

133         <exec_method
134                 type='method'
135                 name='refresh'
136                 exec='/lib/svc/method/net-nwam refresh'
137                 timeout_seconds='60' >
138                 <method_context>
139                         <method_credential user='root' group='root'
140                                 supp_groups='netadm' privileges='zone' />
141                 </method_context>
142         </exec_method>

144         <property_group name='general' type='framework'>
145                 <!-- to start/stop NWAM services -->
146                 <propval name='action_authorization' type='astring'
147                         value='solaris.smf.manage.nwam' />
148                 <propval name='value_authorization' type='astring'
149                         value='solaris.smf.manage.nwam' />
150         </property_group>

152         <property_group name='nwamd' type='application'>
153                 <stability value='Unstable' />
154                 <propval name='debug' type='boolean' value='false' />
155                 <propval name='autoconf' type='boolean' value='false' />
156                 <propval name='ncu_wait_time' type='count' value='60' />
157                 <propval name='condition_check_interval' type='count'
158                         value='120' />
159                 <propval name='scan_interval' type='count' value='120' />
160                 <propval name='scan_level' type='astring' value='weak' />
161                 <propval name='strict_bssid' type='boolean' value='false' />
162                 <propval name='active_ncp' type='astring' value='Automatic' />
163                 <propval name='value_authorization' type='astring'
164                         value='solaris.smf.value.nwam' />
165         </property_group>

167         <template>
168                 <common_name>
169                         <loctext xml:lang='C'>
170                                 physical network interface autoconfiguration
171                         </loctext>
172                 </common_name>
173                 <documentation>
174                         <manpage title='nwamd' section='1M'
175                                 manpath='/usr/share/man' />
176                         <doc_link
177                             name='Network Auto-Magic OpenSolaris Project Page'
178                                 uri='http://hub.opensolaris.org/bin/view/Project
179                                 />
180                 </documentation>
181         </template>

183         </instance>

185         <stability value='Unstable' />

187 </service>

189 </service_bundle>
```

```
**********************************************************
    2142 Tue Jun 12 19:54:51 2012
new/usr/src/lib/libdladm/Makefile
Fixed minor compile errors
**********************************************************
```

```
   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 #
  21 #
  22 # Copyright (c) 2005, 2010, Oracle and/or its affiliates. All rights reserved.
  23 #
  24 #

  26 include $(SRC)/lib/Makefile.lib

  28 HDRS =          libdladm.h libdladm_impl.h libdllink.h libdlaggr.h      \
  29                 libdlwlan.h libdlwlan_impl.h libdlvnic.h libdlvlan.h    \
  30                 libdlmgmt.h libdlflow.h libdlflow_impl.h libdlstat.h    \
  31                 libdlether.h libdlsim.h libdlbridge.h libdliptun.h      \
  32                 libdlib.h

  34 HDRDIR =        common

  36 SUBDIRS =       $(MACH)
  37 $(BUILD64)SUBDIRS += $(MACH64)

  39 POFILE =        libdladm.po
  40 MSGFILES =      common/libdladm.c common/linkprop.c common/secobj.c \
  41                 common/libdllink.c common/libdlaggr.c common/wpa_ie.c   \
  42                 common/wpa_if.c common/libdlwlan.c common/libdlvnic.c   \
  41                 common/libdllink.c common/libdlaggr.c   \
  42                 common/libdlwlan.c common/libdlvnic.c   \
  43                 common/libdlvlan.c common/libdlmgmt.c   \
  44                 common/flowattr.c common/flowprop.c     \
  45                 common/propfuncs.c common/libdlflow.c   \
  46                 common/libdlstat.c common/flowattr.c    \
  47                 common/libdlether.c common/libdlsim.c   \
  48                 common/libdlbridge.c common/libdliptun.c\
  49                 common/libdlib.c

  51 XGETFLAGS =     -a -x libdladm.xcl

  53 all :=          TARGET = all
  54 clean :=        TARGET = clean
  55 clobber :=      TARGET = clobber
  56 install :=      TARGET = install
  57 lint :=         TARGET = lint

  59 .KEEP_STATE:
```



```
  61 all clean clobber install lint: $(SUBDIRS)

  63 install_h:      $(ROOTHDRS)

  65 check:          $(CHECKHDRS)

  67 $(POFILE):      pofile_MSGFILES

  69 _msg:           $(MSGDOMAINPOFILE)

  71 $(SUBDIRS): FRC
  72         @cd $@; pwd; $(MAKE) $(TARGET)

  74 FRC:

  76 include $(SRC)/Makefile.msg.targ
  77 include $(SRC)/lib/Makefile.targ
```

```
**********************************************************
    1630 Tue Jun 12 19:54:52 2012
new/usr/src/lib/libdladm/Makefile.com
Fixed minor compile errors
**********************************************************
```

```
   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 #
  21 #
  22 # Copyright (c) 2005, 2010, Oracle and/or its affiliates. All rights reserved.
  23 #

  25 LIBRARY = libdladm.a
  26 VERS    = .1
  27 OBJECTS = libdladm.o secobj.o linkprop.o libdllink.o libdlaggr.o wpa_ie.o \
  28         wpa_if.o libdlwlan.o libdlvnic.o libdlmgmt.o libdlvlan.o libdlib.o\
  27 OBJECTS = libdladm.o secobj.o linkprop.o libdllink.o libdlaggr.o \
  28         libdlwlan.o libdlvnic.o libdlmgmt.o libdlvlan.o libdlib.o\
  29         flowattr.o flowprop.o propfuncs.o libdlflow.o libdlstat.o \
  30         usage.o libdlether.o libdlsim.o libdlbridge.o libdliptun.o

  32 include ../../Makefile.lib

  34 # install this library in the root filesystem
  35 include ../../Makefile.rootfs

  37 LIBS =          $(DYNLIB) $(LINTLIB)
  38 LDLIBS +=       -ldevinfo -lc -linetutil -lsocket -lscf -lrcm -lnvpair \
  39                 -lexacct -lnsl -lkstat -lcurses -lpool

  41 SRCDIR =        ../common
  42 $(LINTLIB) :=   SRCS = $(SRCDIR)/$(LINTSRC)

  44 CFLAGS +=       $(CCVERBOSE)
  45 CPPFLAGS +=     -I$(SRCDIR) -D_REENTRANT

  47 .KEEP_STATE:

  49 all:            $(LIBS)

  51 lint:           lintcheck

  53 include $(SRC)/lib/Makefile.targ
```

```
**********************************************************
    9129 Tue Jun 12 19:54:53 2012
new/usr/src/lib/libdladm/common/libdladm.h
moved dladm_secobj_class_t to libdladm
**********************************************************
______unchanged_portion_omitted_
```

```
184 typedef enum {
185         DLADM_SECOBJ_CLASS_WEP = 0,
186         DLADM_SECOBJ_CLASS_PSK,
187         DLADM_SECOBJ_CLASS_TLS,
188         DLADM_SECOBJ_CLASS_TTLS,
189         DLADM_SECOBJ_CLASS_PEAP
190 } dladm_secobj_class_t;

192 #endif /* ! codereview */
193 typedef struct {
194         boolean_t       ds_readonly;
195         union {
196                 int     dsu_confid;
197                 nvlist_t *dsu_nvl;
198         } ds_u;
199 } dladm_conf_t;

201 #define ds_confid       ds_u.dsu_confid
202 #define ds_nvl          ds_u.dsu_nvl

204 #define DLADM_INVALID_CONF      0

206 /* opaque dladm handle to libdladm functions */
207 struct dladm_handle;
208 typedef struct dladm_handle *dladm_handle_t;

210 /* open/close handle */
211 extern dladm_status_t   dladm_open(dladm_handle_t *);
212 extern void             dladm_close(dladm_handle_t);

214 /*
215  * retrieve the dld file descriptor from handle, only libdladm and
216  * dlmgmtd are given access to the door file descriptor.
217  */
218 extern int      dladm_dld_fd(dladm_handle_t);

220 typedef struct dladm_arg_info {
221         const char      *ai_name;
222         char            *ai_val[DLADM_MAX_ARG_VALS];
223         uint_t          ai_count;
224 } dladm_arg_info_t;

226 typedef struct dladm_arg_list {
227         dladm_arg_info_t        al_info[DLADM_MAX_ARG_CNT];
228         uint_t                  al_count;
229         char                    *al_buf;
230 } dladm_arg_list_t;

232 typedef enum {
233         DLADM_LOGTYPE_LINK = 1,
234         DLADM_LOGTYPE_FLOW
235 } dladm_logtype_t;

237 typedef struct dladm_usage {
238         char            du_name[MAXLINKNAMELEN];
239         uint64_t        du_duration;
240         uint64_t        du_stime;
241         uint64_t        du_etime;
242         uint64_t        du_ipackets;
```



```
243         uint64_t        du_rbytes;
244         uint64_t        du_opackets;
245         uint64_t        du_obytes;
246         uint64_t        du_bandwidth;
247         boolean_t       du_last;
248 } dladm_usage_t;

250 extern const char       *dladm_status2str(dladm_status_t, char *);
251 extern dladm_status_t   dladm_set_rootdir(const char *);
252 extern const char       *dladm_class2str(datalink_class_t, char *);
253 extern const char       *dladm_media2str(uint32_t, char *);
254 extern uint32_t         dladm_str2media(const char *);
255 extern boolean_t        dladm_valid_linkname(const char *);
256 extern boolean_t        dladm_str2interval(char *, uint32_t *);
257 extern dladm_status_t   dladm_str2bw(char *, uint64_t *);
258 extern const char       *dladm_bw2str(int64_t, char *);
259 extern dladm_status_t   dladm_str2pri(char *, mac_priority_level_t *);
260 extern const char       *dladm_pri2str(mac_priority_level_t, char *);
261 extern dladm_status_t   dladm_str2protect(char *, uint32_t *);
262 extern const char       *dladm_protect2str(uint32_t, char *);
263 extern dladm_status_t   dladm_str2ipv4addr(char *, void *);
264 extern const char       *dladm_ipv4addr2str(void *, char *);
265 extern dladm_status_t   dladm_str2ipv6addr(char *, void *);
266 extern const char       *dladm_ipv6addr2str(void *, char *);

268 extern dladm_status_t   dladm_parse_flow_props(char *, dladm_arg_list_t **,
269                             boolean_t);
270 extern dladm_status_t   dladm_parse_link_props(char *, dladm_arg_list_t **,
271                             boolean_t);
272 extern void             dladm_free_props(dladm_arg_list_t *);
273 extern dladm_status_t   dladm_parse_flow_attrs(char *, dladm_arg_list_t **,
274                             boolean_t);
275 extern void             dladm_free_attrs(dladm_arg_list_t *);

277 extern dladm_status_t   dladm_start_usagelog(dladm_handle_t, dladm_logtype_t,
278                             uint_t);
279 extern dladm_status_t   dladm_stop_usagelog(dladm_handle_t, dladm_logtype_t);
280 extern dladm_status_t   dladm_walk_usage_res(int (*)(dladm_usage_t *, void *),
281                             int, char *, char *, char *, char *, void *);
282 extern dladm_status_t   dladm_walk_usage_time(int (*)(dladm_usage_t *, void *),
283                             int, char *, char *, char *, void *);
284 extern dladm_status_t   dladm_usage_summary(int (*)(dladm_usage_t *, void *),
285                             int, char *, void *);
286 extern dladm_status_t   dladm_usage_dates(int (*)(dladm_usage_t *, void *),
287                             int, char *, char *, void *);
288 extern dladm_status_t   dladm_zone_boot(dladm_handle_t, zoneid_t);
289 extern dladm_status_t   dladm_zone_halt(dladm_handle_t, zoneid_t);

291 extern dladm_status_t   dladm_strs2range(char **, uint_t, mac_propval_type_t,
292                             mac_propval_range_t **);
293 extern dladm_status_t   dladm_range2list(mac_propval_range_t *, void*,
294                             uint_t *);
295 extern int              dladm_range2strs(mac_propval_range_t *, char **);
296 extern dladm_status_t   dladm_list2range(void *, uint_t, mac_propval_type_t,
297                             mac_propval_range_t **);

299 #ifdef  __cplusplus
300 }
301 #endif

303 #endif  /* _LIBDLADM_H */
```

****
7928 Tue Jun 12 19:54:54 2012
new/usr/src/lib/libdladm/common/libdllink.h
secobjs types now are "wep, psk, eap, pin"
dladm_wlan_secmode_t and dladm_secobj_class_t are not related anymore
****
_____***unchanged_portion_omitted_***

```
 65 /*
 66  * Maximum size of secobj value. Note that it should not be greater than
 67  * DLD_SECOBJ_VAL_MAX.
 68  */
 69 #define DLADM_SECOBJ_VAL_MAX    256

 71 /*
 72  * Maximum size of secobj name. Note that it should not be greater than
 73  * DLD_SECOBJ_NAME_MAX.
 74  */
 75 #define DLADM_SECOBJ_NAME_MAX   32

 77 #define DLADM_MAX_PROP_VALCNT   32
 78 /*
 79  * Size of prop_val buffer passed to pd_get function must be at
 80  * least DLADM_PROP_VAL_MAX
 81  */
 82 #define DLADM_PROP_VAL_MAX      128

 84 #define         DLADM_SECOBJ_CLASS_WEP  0
 85 #define         DLADM_SECOBJ_CLASS_WPA  1
 86 typedef int     dladm_secobj_class_t;

 84 typedef int (dladm_walkcb_t)(const char *, void *);

 86 /* possible flags for ma_flags below */
 87 #define DLADM_MACADDR_USED      0x1

 89 typedef enum {
 90         DLADM_HWGRP_TYPE_RX = 0x1,
 91         DLADM_HWGRP_TYPE_TX
 92 } dladm_hwgrp_type_t;
```

_____***unchanged_portion_omitted_***

```
**********************************************************
   45964 Tue Jun 12 19:54:54 2012
new/usr/src/lib/libdladm/common/libdlwlan.c
ess_list ioctl now provides all scan results properties for wpa/libdlwlan
first integration of wpa_s control interface client code
first integration of wpa_s wpa_ie parsing code
**********************************************************
```

```
/*
 * 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.
 */

#include <libintl.h>
#include <stdio.h>
#include <stdlib.h>
#include <unistd.h>
#include <fcntl.h>
#include <stddef.h>
#include <string.h>
#include <stropts.h>
#include <libdevinfo.h>
#include <net/if.h>
#include <net/if_dl.h>
#include <net/if_types.h>
#include <libdlpi.h>
#include <libdllink.h>
#include <libscf.h>
#include <libdlwlan.h>
#include <libdladm_impl.h>
#include <libdlwlan_impl.h>
#include <net/wpa.h>

static int wpa_instance_create(char *ifname);
static int wpa_instance_delete(char *ifname);
static int wpa_network_config(char *ifname, dladm_wlan_attr_t *attrp,
        dladm_wlan_key_t *keys, boolean_t *create_ibss, char *identity);
static dladm_status_t   wpa_instance_create(dladm_handle_t, datalink_id_t,
                            void *);
static dladm_status_t   wpa_instance_delete(dladm_handle_t, datalink_id_t);

static dladm_status_t   do_get_bsstype(dladm_handle_t, datalink_id_t, void *,
                            int);
static dladm_status_t   do_get_essid(dladm_handle_t, datalink_id_t, void *,
                            int);
static dladm_status_t   do_get_bssid(dladm_handle_t, datalink_id_t, void *,
                            int);
```



```
static dladm_status_t   do_get_signal(dladm_handle_t, datalink_id_t, void *,
                            int);
static dladm_status_t   do_get_encryption(dladm_handle_t, datalink_id_t, void *,
                            int);
static dladm_status_t   do_get_authmode(dladm_handle_t, datalink_id_t, void *,
                            int);
static dladm_status_t   do_get_linkstatus(dladm_handle_t, datalink_id_t, void *,
                            int);
static dladm_status_t   do_get_esslist(dladm_handle_t, datalink_id_t, void  *,
                            int);
static dladm_status_t   do_get_rate(dladm_handle_t, datalink_id_t, void *, int);
static dladm_status_t   do_get_mode(dladm_handle_t, datalink_id_t, void *, int);
static dladm_status_t   do_get_capability(dladm_handle_t, datalink_id_t, void *,
                            int);
static dladm_status_t   do_get_wpamode(dladm_handle_t, datalink_id_t, void *,
                            int);

static dladm_status_t   do_set_bsstype(dladm_handle_t, datalink_id_t,
                            dladm_wlan_bsstype_t *);
static dladm_status_t   do_set_authmode(dladm_handle_t, datalink_id_t,
                            dladm_wlan_auth_t *);
static dladm_status_t   do_set_encryption(dladm_handle_t, datalink_id_t,
                            dladm_wlan_secmode_t *);
static dladm_status_t   do_set_essid(dladm_handle_t, datalink_id_t,
                            dladm_wlan_essid_t *);
static dladm_status_t   do_set_createibss(dladm_handle_t, datalink_id_t,
                            boolean_t *);
static dladm_status_t   do_set_key(dladm_handle_t, datalink_id_t,
                            dladm_wlan_key_t *, uint_t);
static dladm_status_t   do_set_channel(dladm_handle_t, datalink_id_t,
                            dladm_wlan_channel_t *);

static dladm_status_t   do_scan(dladm_handle_t, datalink_id_t, void *, int);
static dladm_status_t   do_connect(dladm_handle_t, datalink_id_t, void *, int,
                            dladm_wlan_attr_t *, boolean_t, void *, uint_t,
                            int);
static dladm_status_t   do_disconnect(dladm_handle_t, datalink_id_t, void *,
                            int);
static boolean_t        find_val_by_name(const char *, val_desc_t *,
                            uint_t, uint_t *);
static boolean_t        find_name_by_val(uint_t, val_desc_t *, uint_t, char **);
static void             generate_essid(dladm_wlan_essid_t *);

static dladm_status_t   dladm_wlan_wlresult2status(wldp_t *);
static dladm_status_t   dladm_wlan_validate(dladm_handle_t, datalink_id_t);

static val_desc_t       linkstatus_vals[] = {
        { "disconnected", DLADM_WLAN_LINK_DISCONNECTED  },
        { "connected",    DLADM_WLAN_LINK_CONNECTED     }
};

static val_desc_t       secmode_vals[] = {
        { "none",       DLADM_WLAN_SECMODE_NONE },
        { "wep",        DLADM_WLAN_SECMODE_WEP  },
        { "wpa-psk",    DLADM_WLAN_SECMODE_PSK  },
        { "wpa-eap",    DLADM_WLAN_SECMODE_EAP  }
        { "wpa",        DLADM_WLAN_SECMODE_WPA          }
};
_____unchanged_portion_omitted_

/* capital chars*/
#endif /* ! codereview */
static val_desc_t       auth_vals[] = {
        { "OPEN",       DLADM_WLAN_AUTH_OPEN            },
        { "SHARED",     DLADM_WLAN_AUTH_SHARED          }
        { "open",       DLADM_WLAN_AUTH_OPEN            },
```

```
130         { "shared",     DLADM_WLAN_AUTH_SHARED          }
109 };

111 static val_desc_t      bsstype_vals[] = {
112         { "bss",        DLADM_WLAN_BSSTYPE_BSS          },
113         { "ibss",       DLADM_WLAN_BSSTYPE_IBSS         }
135         { "ibss",       DLADM_WLAN_BSSTYPE_IBSS         },
136         { "any",        DLADM_WLAN_BSSTYPE_ANY          }
114 };

139 #define WLDP_BUFSIZE (MAX_BUF_LEN - WIFI_BUF_OFFSET)

116 static dladm_status_t
117 dladm_wlan_wlresult2status(wldp_t *gbuf)
118 {
119         switch (gbuf->wldp_result) {
120         case WL_SUCCESS:
121                 return (DLADM_STATUS_OK);

123         case WL_NOTSUPPORTED:
124         case WL_LACK_FEATURE:
125                 return (DLADM_STATUS_NOTSUP);

127         case WL_READONLY:
128                 return (DLADM_STATUS_PROPRDONLY);

130         default:
131                 break;
132         }

134         return (DLADM_STATUS_FAILED);
135 }
______unchanged_portion_omitted_

208 #define IEEE80211_RATE  0x7f
183 static void
184 fill_wlan_attr(wl_ess_conf_t *wlp, dladm_wlan_attr_t *attrp)
185 {
186         int i;

188         (void) memset(attrp, 0, sizeof (*attrp));

190         attrp->wa_essid.we_length = wlp->wl_ess_conf_essid.wl_essid_length;
191         (void) memcpy(attrp->wa_essid.we_bytes, wlp->wl_ess_conf_essid.wl_essid_
192             attrp->wa_essid.we_length);
216         (void) snprintf(attrp->wa_essid.we_bytes, DLADM_WLAN_MAX_ESSID_LEN,
217             "%s", wlp->wl_ess_conf_essid.wl_essid_essid);
193         attrp->wa_valid |= DLADM_WLAN_ATTR_ESSID;

195         /*
196          * Note: SSID is an array of octets, i.e.,
197          * it is not nul terminated and can, at least in theory,
198          * contain control characters (including nul) and as such,
199          * should be processed as binary data, not a printable string.
200          *
201          *the following call has been replaced with the one above:
202          *(void) snprintf(attrp->wa_essid.we_bytes, DLADM_WLAN_MAX_ESSID_LEN,
203          *   "%s", wlp->wl_ess_conf_essid.wl_essid_essid);
204         */

206 #endif /* ! codereview */
207         (void) memcpy(attrp->wa_bssid.wb_bytes, wlp->wl_ess_conf_bssid,
208             DLADM_WLAN_BSSID_LEN);
209         attrp->wa_valid |= DLADM_WLAN_ATTR_BSSID;

211         /*
```



```
212          * Open/Shared is simply not present in the AP's beacon.  It is a
213          * deficiency of WEP, which WPA fixed by only allowing open
214          * authentication for WPA connections. There is simply no way to tell
215          * whether an AP is using shared auth or open system auth because a WEP
216          * AP never broadcasts that information.
217          * [net80211 sets it to 1 (OPEN) by default for scanned nodes]
218          *
219          * If wpa_ie is not present and IEEE80211_CAP_PRIVACY is on => WEP
220          * If wpa_ie is not present and IEEE80211_CAP_PRIVACY is off => NONE
221          */

223         if (wlp->wl_ess_conf_wpa_ie_len == 0) {
224                 if (wlp->wl_ess_conf_caps & IEEE80211_CAP_PRIVACY) {
225                         attrp->wa_auth = DLADM_WLAN_AUTH_SHARED;
226                         attrp->wa_secmode = DLADM_WLAN_SECMODE_WEP;
227                 } else {
228                         attrp->wa_auth = DLADM_WLAN_AUTH_NONE;
229                         attrp->wa_secmode = DLADM_WLAN_SECMODE_NONE;
230                 }
231         } else {
232                 attrp->wa_auth = DLADM_WLAN_AUTH_OPEN;
233                 wpa_parse_wpa_ie(wlp->wl_ess_conf_wpa_ie,
234                     wlp->wl_ess_conf_wpa_ie_len, &attrp->wa_ie);
235                 /*
236                  * check wpa_ie.c defines
237                  */
238                 if (attrp->wa_ie.key_mgmt == 0)
239                         attrp->wa_secmode = DLADM_WLAN_SECMODE_EAP;
240                 if (attrp->wa_ie.key_mgmt == 2)
241                         attrp->wa_secmode = DLADM_WLAN_SECMODE_PSK;
242         }
243         attrp->wa_valid |= DLADM_WLAN_ATTR_AUTH;
220         attrp->wa_secmode = (wlp->wl_ess_conf_wepenabled ==
221             WL_ENC_WEP ? DLADM_WLAN_SECMODE_WEP : DLADM_WLAN_SECMODE_NONE);
222         if (wlp->wl_ess_conf_reserved[0] > 0)
223                 attrp->wa_secmode = DLADM_WLAN_SECMODE_WPA;
244         attrp->wa_valid |= DLADM_WLAN_ATTR_SECMODE;

246         /* caps values are not mutually-exclusive here */
247         if (wlp->wl_ess_conf_caps & IEEE80211_CAP_IBSS)
248                 attrp->wa_bsstype = DLADM_WLAN_BSSTYPE_IBSS;
249         if (wlp->wl_ess_conf_caps & IEEE80211_CAP_ESS)
250                 attrp->wa_bsstype = DLADM_WLAN_BSSTYPE_BSS;
226         attrp->wa_bsstype = (wlp->wl_ess_conf_bsstype == WL_BSS_BSS ?
227             DLADM_WLAN_BSSTYPE_BSS : DLADM_WLAN_BSSTYPE_IBSS);
251         attrp->wa_valid |= DLADM_WLAN_ATTR_BSSTYPE;

230         attrp->wa_auth = (wlp->wl_ess_conf_authmode == 0 ?
231             DLADM_WLAN_AUTH_OPEN : DLADM_WLAN_AUTH_SHARED);
232         attrp->wa_valid |= DLADM_WLAN_ATTR_AUTH;

253         attrp->wa_strength = DLADM_WLAN_SIGNAL2STRENGTH(wlp->wl_ess_conf_sl);
254         attrp->wa_valid |= DLADM_WLAN_ATTR_STRENGTH;

256         attrp->wa_mode = do_convert_mode((wl_phy_conf_t *)&wlp->wl_phy_conf);
257         attrp->wa_valid |= DLADM_WLAN_ATTR_MODE;

259         for (i = 0; i < MAX_SCAN_SUPPORT_RATES; i++) {
260                 wlp->wl_supported_rates[i] &= IEEE80211_RATE;
261                 if (wlp->wl_supported_rates[i] > attrp->wa_speed)
262                         attrp->wa_speed = wlp->wl_supported_rates[i];
263         }
264         if (attrp->wa_speed > 0)
265                 attrp->wa_valid |= DLADM_WLAN_ATTR_SPEED;

267         if (i_dladm_wlan_convert_chan((wl_phy_conf_t *)&wlp->wl_phy_conf,
```

```
268             &attrp->wa_channel))
269                 attrp->wa_valid |= DLADM_WLAN_ATTR_CHANNEL;
270 }

272 #define WLDP_BUFSIZE    (MAX_BUF_LEN - WIFI_BUF_OFFSET)

274 #endif /* ! codereview */
275 dladm_status_t
276 dladm_wlan_scan(dladm_handle_t handle, datalink_id_t linkid, void *arg,
277     boolean_t (*func)(void *, dladm_wlan_attr_t *))
278 {
279         int                     i;
280         uint32_t                count;
281         wl_ess_conf_t           *wlp;
282         wl_ess_list_t           *wls = NULL;
283         char                    buf[WLDP_BUFSIZE];
284         wl_linkstatus_t         wl_status;
285         dladm_wlan_attr_t       wlattr;
286         dladm_status_t          status;

288         if ((status = dladm_wlan_validate(handle, linkid)) != DLADM_STATUS_OK)
289                 goto done;

291         status = do_get_linkstatus(handle, linkid, &wl_status,
292             sizeof (wl_status));
293         if (status != DLADM_STATUS_OK)
294                 goto done;

296         if ((status = do_scan(handle, linkid, buf, sizeof (buf))) !=
297             DLADM_STATUS_OK)
298                 goto done;

300         if (func == NULL) {
301                 status = DLADM_STATUS_OK;
302                 goto done;
303         }

305         wls = malloc(WLDP_BUFSIZE);
306         if (wls == NULL) {
307                 status = DLADM_STATUS_NOMEM;
308                 goto done;
309         }

311         if ((status = dladm_wlan_get_esslist(handle, linkid, wls, WLDP_BUFSIZE))
253         if ((status = do_get_esslist(handle, linkid, wls, WLDP_BUFSIZE))
312             != DLADM_STATUS_OK)
313                 goto done;

315         wlp = wls->wl_ess_list_ess;
316         count = wls->wl_ess_list_num;

318         for (i = 0; i < count; i++, wlp++) {
319                 fill_wlan_attr(wlp, &wlattr);
320                 if (!func(arg, &wlattr))
321                         break;
322         }

324         if (wl_status != WL_CONNECTED) {
325                 status = do_get_linkstatus(handle, linkid, &wl_status,
326                     sizeof (&wl_status));
327                 if (status != DLADM_STATUS_OK)
328                         goto done;
329                 if (wl_status == WL_CONNECTED)
330                         (void) do_disconnect(handle, linkid, buf, sizeof (buf));
331         }
```



```
333         status = DLADM_STATUS_OK;
334 done:
335         free(wls);
336         return (status);
337 }
_____unchanged_portion_omitted_

380 /*
381  * Callback function used by dladm_wlan_connect() to filter out unwanted
382  * WLANs when scanning for available WLANs.  Always returns B_TRUE to
383  * continue the scan.
384  */
385 static boolean_t
386 connect_cb(void *arg, dladm_wlan_attr_t *attrp)
387 {
388         attr_node_t             *nodep;
389         dladm_wlan_attr_t       *fattrp;
390         connect_state_t         *statep = (connect_state_t *)arg;

392         fattrp = statep->cs_attr;
393         if (fattrp == NULL)
394                 goto append;

396         if ((fattrp->wa_valid & attrp->wa_valid) != fattrp->wa_valid)
397                 return (B_TRUE);

399         if ((fattrp->wa_valid & DLADM_WLAN_ATTR_ESSID) != 0 &&
400             memcmp(fattrp->wa_essid.we_bytes, attrp->wa_essid.we_bytes,
401             attrp->wa_essid.we_length) != 0)
342             strncmp(fattrp->wa_essid.we_bytes, attrp->wa_essid.we_bytes,
343             DLADM_WLAN_MAX_ESSID_LEN) != 0)
402                 return (B_TRUE);

404         if ((fattrp->wa_valid & DLADM_WLAN_ATTR_SECMODE) != 0 &&
405             fattrp->wa_secmode != attrp->wa_secmode)
406                 return (B_TRUE);

408         if ((fattrp->wa_valid & DLADM_WLAN_ATTR_MODE) != 0 &&
409             fattrp->wa_mode != attrp->wa_mode)
410                 return (B_TRUE);

412         if ((fattrp->wa_valid & DLADM_WLAN_ATTR_STRENGTH) != 0 &&
413             fattrp->wa_strength != attrp->wa_strength)
414                 return (B_TRUE);

416         if ((fattrp->wa_valid & DLADM_WLAN_ATTR_SPEED) != 0 &&
417             fattrp->wa_speed != attrp->wa_speed)
418                 return (B_TRUE);

420         if ((fattrp->wa_valid & DLADM_WLAN_ATTR_AUTH) != 0) {
421                 attrp->wa_auth = fattrp->wa_auth;
422                 attrp->wa_valid |= DLADM_WLAN_ATTR_AUTH;
423         }

425         if ((fattrp->wa_valid & DLADM_WLAN_ATTR_BSSTYPE) != 0 &&
426             fattrp->wa_bsstype != attrp->wa_bsstype)
427                 return (B_TRUE);

429         if ((fattrp->wa_valid & DLADM_WLAN_ATTR_BSSID) != 0 &&
430             memcmp(fattrp->wa_bssid.wb_bytes, attrp->wa_bssid.wb_bytes,
431             DLADM_WLAN_BSSID_LEN) != 0)
432                 return (B_TRUE);
433 append:
434         nodep = malloc(sizeof (attr_node_t));
435         if (nodep == NULL)
436                 return (B_TRUE);
```

```
438         (void) memcpy(&nodep->an_attr, attrp, sizeof (dladm_wlan_attr_t));
439         nodep->an_next = statep->cs_list;
440         statep->cs_list = nodep;
441         statep->cs_count++;

443         return (B_TRUE);
444 }
```

```
388 #define IEEE80211_C_WPA         0x01800000

390 static dladm_status_t
391 do_connect(dladm_handle_t handle, datalink_id_t linkid, void *buf, int bufsize,
392     dladm_wlan_attr_t *attrp, boolean_t create_ibss, void *keys,
393     uint_t key_count, int timeout)
394 {
395         dladm_wlan_secmode_t    secmode;
396         dladm_wlan_auth_t       authmode;
397         dladm_wlan_bsstype_t    bsstype;
398         dladm_wlan_essid_t      essid;
399         boolean_t               essid_valid = B_FALSE;
400         dladm_status_t          status;
401         dladm_wlan_channel_t    channel;
402         hrtime_t                start;
403         wl_capability_t         *caps;
404         wl_linkstatus_t         wl_status;

406         if ((attrp->wa_valid & DLADM_WLAN_ATTR_CHANNEL) != 0) {
407                 channel = attrp->wa_channel;
408                 status = do_set_channel(handle, linkid, &channel);
409                 if (status != DLADM_STATUS_OK)
410                         goto fail;
411         }

413         secmode = ((attrp->wa_valid & DLADM_WLAN_ATTR_SECMODE) != 0) ?
414             attrp->wa_secmode : DLADM_WLAN_SECMODE_NONE;

416         if ((status = do_set_encryption(handle, linkid, &secmode)) !=
417             DLADM_STATUS_OK)
418                 goto fail;

420         authmode = ((attrp->wa_valid & DLADM_WLAN_ATTR_AUTH) != 0) ?
421             attrp->wa_auth : DLADM_WLAN_AUTH_OPEN;

423         if ((status = do_set_authmode(handle, linkid, &authmode)) !=
424             DLADM_STATUS_OK)
425                 goto fail;

427         bsstype = ((attrp->wa_valid & DLADM_WLAN_ATTR_BSSTYPE) != 0) ?
428             attrp->wa_bsstype : DLADM_WLAN_BSSTYPE_BSS;

430         if ((status = do_set_bsstype(handle, linkid, &bsstype)) !=
431             DLADM_STATUS_OK)
432                 goto fail;

434         if (secmode == DLADM_WLAN_SECMODE_WEP) {
435                 if (keys == NULL || key_count == 0 ||
436                     key_count > MAX_NWEPKEYS) {
437                         status = DLADM_STATUS_BADARG;
438                         goto fail;
439                 }
440                 status = do_set_key(handle, linkid, keys, key_count);
441                 if (status != DLADM_STATUS_OK)
442                         goto fail;
443         } else if (secmode == DLADM_WLAN_SECMODE_WPA) {
444                 if (keys == NULL || key_count == 0 ||
```



```
445                     key_count > MAX_NWEPKEYS) {
446                         status = DLADM_STATUS_BADARG;
447                         goto fail;
448                 }
449                 status = do_get_capability(handle, linkid, buf, bufsize);
450                 if (status != DLADM_STATUS_OK)
451                         goto fail;
452                 caps = (wl_capability_t *)buf;
453                 if ((caps->caps & IEEE80211_C_WPA) == 0)
454                         return (DLADM_STATUS_NOTSUP);
455         }

457         if (create_ibss) {
458                 status = do_set_channel(handle, linkid, &channel);
459                 if (status != DLADM_STATUS_OK)
460                         goto fail;

462                 status = do_set_createibss(handle, linkid, &create_ibss);
463                 if (status != DLADM_STATUS_OK)
464                         goto fail;

466                 if ((attrp->wa_valid & DLADM_WLAN_ATTR_ESSID) == 0) {
467                         generate_essid(&essid);
468                         essid_valid = B_TRUE;
469                 }
470         }

472         if ((attrp->wa_valid & DLADM_WLAN_ATTR_ESSID) != 0) {
473                 essid = attrp->wa_essid;
474                 essid_valid = B_TRUE;
475         }

477         if (!essid_valid) {
478                 status = DLADM_STATUS_FAILED;
479                 goto fail;
480         }

482         if ((status = do_set_essid(handle, linkid, &essid)) != DLADM_STATUS_OK)
483                 goto fail;

485         /*
486          * Because wpa daemon needs getting essid from driver,
487          * we need call do_set_essid() first, then call wpa_instance_create().
488          */
489         if (secmode == DLADM_WLAN_SECMODE_WPA && keys != NULL)
490                 (void) wpa_instance_create(handle, linkid, keys);

492         start = gethrtime();
493         for (;;) {
494                 status = do_get_linkstatus(handle, linkid, &wl_status,
495                     sizeof (wl_status));
496                 if (status != DLADM_STATUS_OK)
497                         goto fail;

499                 if (wl_status == WL_CONNECTED)
500                         break;

502                 (void) poll(NULL, 0, DLADM_WLAN_CONNECT_POLLRATE);
503                 if ((timeout >= 0) && (gethrtime() - start) /
504                     NANOSEC >= timeout) {
505                         status = DLADM_STATUS_TIMEDOUT;
506                         goto fail;
507                 }
508         }
509         status = DLADM_STATUS_OK;
510 fail:
```

```
511         return (status);
512 }

446 dladm_status_t
447 dladm_wlan_connect(dladm_handle_t handle, datalink_id_t linkid,
448     dladm_wlan_attr_t *attrp, int timeout, void *keys,
449     uint_t key_count, uint_t flags, char *identity)
516     dladm_wlan_attr_t *attrp, int timeout, void *keys, uint_t key_count,
517     uint_t flags)
450 {
451         char                    ifname[MAXLINKNAMELEN];
452         boolean_t               create_ibss;
519         int                     i;
520         char                    buf[WLDP_BUFSIZE];
521         connect_state_t         state = {0, NULL, NULL};
522         attr_node_t             *nodep = NULL;
523         boolean_t               create_ibss, set_authmode;
524         dladm_wlan_attr_t       **wl_list = NULL;
453         dladm_status_t          status;
454         wl_linkstatus_t         wl_status;
455         hrtime_t                start;
456 #endif /* ! codereview */

458         if ((status = dladm_wlan_validate(handle, linkid)) != DLADM_STATUS_OK)
459                 return (status);

461         if ((status = dladm_datalink_id2info(handle, linkid, NULL, NULL, NULL,
462             ifname, sizeof (ifname))) != DLADM_STATUS_OK)
463                 goto done;

465 #endif /* ! codereview */
466         if ((status = do_get_linkstatus(handle, linkid, &wl_status,
467             sizeof (wl_status))) != DLADM_STATUS_OK)
468                 goto done;

470         if (wl_status == WL_CONNECTED) {
471                 status = DLADM_STATUS_ISCONN;
472                 goto done;
473         }

475         if (wpa_instance_create(ifname))
476                 goto done;

478         if (wpa_network_config(ifname, attrp, keys, &create_ibss, identity)) {
479                 status = DLADM_STATUS_BADVAL;
480                 goto done;
481         }

483         /*
484          * DLADM_WLAN_CONNECT_CREATEIBSS:
485          * If this flag is set and the bsstype attribute attr->wa_bsstype is
486          * set to DLADM_WLAN_BSSTYPE_IBSS:
487          * If the essid attribute attr->wa_essid is specified and there exists
488          * no WLAN on the discovered WLAN with this particular essid, an adhoc
489          * WLAN with essid equal to attr->wa_essid will be created.
490          * If the essid attribute is not specified, an adhoc WLAN with a random
491          * essid will be created, irrespective of which WLANs are available.
492          */

494         /* we will manage this inside wpa_network config
527         set_authmode = ((attrp != NULL) &&
528             (attrp->wa_valid & DLADM_WLAN_ATTR_MODE) != 0);
495         create_ibss = ((flags & DLADM_WLAN_CONNECT_CREATEIBSS) != 0 &&
496             attrp != NULL && (attrp->wa_valid & DLADM_WLAN_ATTR_BSSTYPE) != 0 &&
530             attrp != NULL &&
531             (attrp->wa_valid & DLADM_WLAN_ATTR_BSSTYPE) != 0 &&
```



```
497             attrp->wa_bsstype == DLADM_WLAN_BSSTYPE_IBSS);

499         if ((create_ibss && attrp != NULL &&
534         if ((flags & DLADM_WLAN_CONNECT_NOSCAN) != 0 ||
535             (create_ibss && attrp != NULL &&
500             (attrp->wa_valid & DLADM_WLAN_ATTR_ESSID) == 0)) {
501                 status = do_connect(handle, linkid, buf, sizeof (buf), attrp,
502                     create_ibss, keys, key_count, timeout);
503                 goto done;
540         }

542         state.cs_attr = attrp;
543         state.cs_list = NULL;
544         state.cs_count = 0;

546         status = dladm_wlan_scan(handle, linkid, &state, connect_cb);
547         if (status != DLADM_STATUS_OK)
548                 goto done;

505         if (state.cs_count == 0) {
506                 if (!create_ibss) {
507                         status = DLADM_STATUS_NOTFOUND;
508                         goto done;
509                 }
510         } */
555                 status = do_connect(handle, linkid, buf, sizeof (buf),
556                     attrp, create_ibss, keys, key_count, timeout);
557                 goto done;
558         }

512         start = gethrtime();
513         for (;;) {
514                 status = do_get_linkstatus(handle, linkid, &wl_status,
515                     sizeof (wl_status));
516                 if (status != DLADM_STATUS_OK)
560         wl_list = malloc(state.cs_count * sizeof (dladm_wlan_attr_t *));
561         if (wl_list == NULL) {
562                 status = DLADM_STATUS_NOMEM;
517                         goto done;
564         }

519                 if (wl_status == WL_CONNECTED)
566         nodep = state.cs_list;
567         for (i = 0; i < state.cs_count; i++) {
568                 wl_list[i] = &nodep->an_attr;
569                 nodep = nodep->an_next;
570         }
571         qsort(wl_list, state.cs_count, sizeof (dladm_wlan_attr_t *),
572             attr_compare);

574         for (i = 0; i < state.cs_count; i++) {
575                 dladm_wlan_attr_t       *ap = wl_list[i];

577                 status = do_connect(handle, linkid, buf, sizeof (buf),
578                     ap, create_ibss, keys, key_count, timeout);
579                 if (status == DLADM_STATUS_OK)
580                         break;

582                 if (!set_authmode) {
583                         ap->wa_auth = DLADM_WLAN_AUTH_SHARED;
584                         ap->wa_valid |= DLADM_WLAN_ATTR_AUTH;
585                         status = do_connect(handle, linkid, buf, sizeof (buf),
586                             ap, create_ibss, keys, key_count, timeout);
587                         if (status == DLADM_STATUS_OK)
520                         break;
```

```
522                 (void) poll(NULL, 0, DLADM_WLAN_CONNECT_POLLRATE);
523                 if ((timeout >= 0) && (gethrtime() - start) /
524                     NANOSEC >= timeout) {
525                         status = DLADM_STATUS_TIMEDOUT;
526                         goto done;
527 #endif /* ! codereview */
528                 }
529         }

531 #endif /* ! codereview */
532 done:
533         if ((status != DLADM_STATUS_OK) && (status != DLADM_STATUS_ISCONN))
534                 (void) do_disconnect(handle, linkid, NULL, NULL);
589                 (void) do_disconnect(handle, linkid, buf, sizeof (buf));

591         while (state.cs_list != NULL) {
592                 nodep = state.cs_list;
593                 state.cs_list = nodep->an_next;
594                 free(nodep);
595         }
596         free(wl_list);
536         return (status);
537 }
_____unchanged_portion_omitted_

576 dladm_status_t
577 dladm_wlan_get_linkattr(dladm_handle_t handle, datalink_id_t linkid,
578     dladm_wlan_linkattr_t *attrp)
579 {
580         wl_rssi_t               signal;
581         wl_bss_type_t           bsstype;
582         wl_authmode_t           authmode;
583         /* used only in dladm show-wifi */
584 #endif /* ! codereview */
585         wl_encryption_t         encryption;
586         wl_rates_t              *ratesp = NULL;
587         dladm_wlan_attr_t       *wl_attrp;
588         dladm_status_t          status;
589         wl_bssid_t              wl_bssid;
644         char                    buf[WLDP_BUFSIZE];
590         wl_essid_t              wls;
591         wl_phy_conf_t           wl_phy_conf;
592         wl_linkstatus_t         wl_status;

594         if (attrp == NULL)
595                 return (DLADM_STATUS_BADARG);

597         if ((status = dladm_wlan_validate(handle, linkid)) != DLADM_STATUS_OK)
598                 goto done;

600         (void) memset(attrp, 0, sizeof (*attrp));
601         wl_attrp = &attrp->la_wlan_attr;

603         if ((status = do_get_linkstatus(handle, linkid, &wl_status,
604             sizeof (wl_status))) != DLADM_STATUS_OK)
605                 goto done;

607         attrp->la_valid |= DLADM_WLAN_LINKATTR_STATUS;
608         if (wl_status != WL_CONNECTED)
609                 attrp->la_status = DLADM_WLAN_LINK_DISCONNECTED;
610         else
611                 attrp->la_status = DLADM_WLAN_LINK_CONNECTED;

613         /* essid */
614         if ((status = dladm_wlan_get_essid(handle, linkid, &wls, sizeof (wls)))
668         if ((status = do_get_essid(handle, linkid, &wls, sizeof (wls)))
```



```
615             != DLADM_STATUS_OK)
616                 goto done;
617         wl_attrp->wa_essid.we_length = wls.wl_essid_length;
618         (void) memcpy(wl_attrp->wa_essid.we_bytes, wls.wl_essid_essid,
619             wls.wl_essid_length);

672         (void) strlcpy(wl_attrp->wa_essid.we_bytes, wls.wl_essid_essid,
673             DLADM_WLAN_MAX_ESSID_LEN);

620         wl_attrp->wa_valid |= DLADM_WLAN_ATTR_ESSID;

622         /* bssid */
623         if ((status = dladm_wlan_get_bssid(handle, linkid, wl_bssid, sizeof (wl_
677         if ((status = do_get_bssid(handle, linkid, buf, sizeof (buf)))
624             != DLADM_STATUS_OK)
625                 goto done;
626         (void) memcpy(wl_attrp->wa_bssid.wb_bytes, wl_bssid, DLADM_WLAN_BSSID_LE

681         (void) memcpy(wl_attrp->wa_bssid.wb_bytes, buf, DLADM_WLAN_BSSID_LEN);

627         wl_attrp->wa_valid |= DLADM_WLAN_ATTR_BSSID;

629         if (attrp->la_status == DLADM_WLAN_LINK_DISCONNECTED) {
630                 attrp->la_valid |= DLADM_WLAN_LINKATTR_WLAN;
631                 status = DLADM_STATUS_OK;
632                 goto done;
633         }

635         /* returned values used only in dladm show-wifi */
636 #endif /* ! codereview */
637         if ((status = do_get_encryption(handle, linkid, &encryption,
638             sizeof (encryption))) != DLADM_STATUS_OK)
639                 goto done;

641         wl_attrp->wa_valid |= DLADM_WLAN_ATTR_SECMODE;

643         /* !!!! FIX ME!!!! */
644         /*
645          *  this should be alligned with protocol/scan results,
646          * not types of keys used
647          */
648 #endif /* ! codereview */
649         switch (encryption) {
650         case WL_NOENCRYPTION:
651                 wl_attrp->wa_secmode = DLADM_WLAN_SECMODE_NONE;
652                 break;
653         case WL_ENC_WEP:
654                 wl_attrp->wa_secmode = DLADM_WLAN_SECMODE_WEP;
655                 break;
656         case WL_ENC_WPA:
657                 wl_attrp->wa_secmode = DLADM_WLAN_SECMODE_PSK;
691                 wl_attrp->wa_secmode = DLADM_WLAN_SECMODE_WPA;
658                 break;
659         default:
660                 wl_attrp->wa_valid &= ~DLADM_WLAN_ATTR_SECMODE;
661                 break;
662         }

664         if ((status = do_get_signal(handle, linkid, &signal, sizeof (signal)))
665             != DLADM_STATUS_OK)
666                 goto done;

668         wl_attrp->wa_valid |= DLADM_WLAN_ATTR_STRENGTH;
669         wl_attrp->wa_strength = DLADM_WLAN_SIGNAL2STRENGTH(signal);

671         ratesp = malloc(WLDP_BUFSIZE);
```

```
672         if (ratesp == NULL) {
673                 status = DLADM_STATUS_NOMEM;
674                 goto done;
675         }

677         if ((status = do_get_rate(handle, linkid, ratesp, WLDP_BUFSIZE))
678             != DLADM_STATUS_OK)
679                 goto done;

681         if (ratesp->wl_rates_num > 0) {
682                 uint_t  i, r = 0;

684                 for (i = 0; i < ratesp->wl_rates_num; i++) {
685                         if (ratesp->wl_rates_rates[i] > r)
686                                 r = ratesp->wl_rates_rates[i];
687                 }
688                 wl_attrp->wa_speed = r;
689                 wl_attrp->wa_valid |= DLADM_WLAN_ATTR_SPEED;
690         }

692         if ((status = do_get_authmode(handle, linkid, &authmode,
693             sizeof (authmode))) != DLADM_STATUS_OK)
694                 goto done;

696         wl_attrp->wa_valid |= DLADM_WLAN_ATTR_AUTH;

698         switch (authmode) {
699         case WL_OPENSYSTEM:
700                 wl_attrp->wa_auth = DLADM_WLAN_AUTH_OPEN;
701                 break;
702         case WL_SHAREDKEY:
703                 wl_attrp->wa_auth = DLADM_WLAN_AUTH_SHARED;
704                 break;
705         default:
706                 wl_attrp->wa_valid &= ~DLADM_WLAN_ATTR_AUTH;
707                 break;
708         }

710         if ((status = do_get_bsstype(handle, linkid, &bsstype,
711             sizeof (bsstype))) != DLADM_STATUS_OK)
712                 goto done;

714         wl_attrp->wa_valid |= DLADM_WLAN_ATTR_BSSTYPE;

716         switch (bsstype) {
717         case WL_BSS_BSS:
718                 wl_attrp->wa_bsstype = DLADM_WLAN_BSSTYPE_BSS;
719                 break;
720         case WL_BSS_IBSS:
721                 wl_attrp->wa_bsstype = DLADM_WLAN_BSSTYPE_IBSS;
722                 break;
757         case WL_BSS_ANY:
758                 wl_attrp->wa_bsstype = DLADM_WLAN_BSSTYPE_ANY;
759                 break;
723         default:
724                 wl_attrp->wa_valid &= ~DLADM_WLAN_ATTR_BSSTYPE;
725                 break;
726         }

728         if ((status = do_get_mode(handle, linkid, &wl_phy_conf,
729             sizeof (wl_phy_conf))) != DLADM_STATUS_OK)
730                 goto done;

732         wl_attrp->wa_mode = do_convert_mode(&wl_phy_conf);
733         wl_attrp->wa_valid |= DLADM_WLAN_ATTR_MODE;
734         if (wl_attrp->wa_mode != DLADM_WLAN_MODE_NONE)
```



```
735                 wl_attrp->wa_valid |= DLADM_WLAN_ATTR_MODE;

737         attrp->la_valid |= DLADM_WLAN_LINKATTR_WLAN;
738         status = DLADM_STATUS_OK;

740 done:
741         free(ratesp);
742         return (status);
743 }
```

*unchanged_portion_omitted_*

```
791 /* data to string */

793 #endif /* ! codereview */
794 const char *
795 dladm_wlan_essid2str(dladm_wlan_essid_t *essid, char *buf)
796 {
797         (void) snprintf(buf, essid->we_length+1, "%s", essid->we_bytes);
828         (void) snprintf(buf, DLADM_STRSIZE, "%s", essid->we_bytes);
798         return (buf);
799 }
```

*unchanged_portion_omitted_*

```
820 /*
821  * SECMODE SHOULD NEVER BE PRINTED, in known-wlan.conf and secobj.conf we'll use
822  * secobj_class values, and for scan results wpa_s values
823  */
824 #endif /* ! codereview */
825 const char *
826 dladm_wlan_secmode2str(dladm_wlan_secmode_t *secmode, char *buf)
827 {
828         return (dladm_wlan_val2str((uint_t)*secmode, secmode_vals,
829             VALCNT(secmode_vals), buf));
830 }

832 const char *
833 dladm_wlan_strength2str(dladm_wlan_strength_t *strength, char *buf)
834 {
835         return (dladm_wlan_val2str((uint_t)*strength, strength_vals,
836             VALCNT(strength_vals), buf));
837 }

839 const char *
840 dladm_wlan_mode2str(dladm_wlan_mode_t *mode, char *buf)
841 {
842         return (dladm_wlan_val2str((uint_t)*mode, mode_vals,
843             VALCNT(mode_vals), buf));
844 }

846 const char *
847 dladm_wlan_speed2str(dladm_wlan_speed_t *speed, char *buf)
848 {
849         (void) snprintf(buf, DLADM_STRSIZE, "%.*f", *speed % 2,
850             (float)(*speed) / 2);
851         return (buf);
852 }

854 const char *
855 dladm_wlan_auth2str(dladm_wlan_auth_t *auth, char *buf)
856 {
857         return (dladm_wlan_val2str((uint_t)*auth, auth_vals,
858             VALCNT(auth_vals), buf));
859 }

861 const char *
862 dladm_wlan_bsstype2str(dladm_wlan_bsstype_t *bsstype, char *buf)
```

```
 863 {
 864         return (dladm_wlan_val2str((uint_t)*bsstype, bsstype_vals,
 865             VALCNT(bsstype_vals), buf));
 866 }

 868 const char *
 869 dladm_wlan_linkstatus2str(dladm_wlan_linkstatus_t *linkstatus, char *buf)
 870 {
 871         return (dladm_wlan_val2str((uint_t)*linkstatus, linkstatus_vals,
 872             VALCNT(linkstatus_vals), buf));
 873 }

 875 /* string to data */

 877 #endif /* ! codereview */
 878 dladm_status_t
 879 dladm_wlan_str2essid(const char *str, dladm_wlan_essid_t *essid)
 880 {
 881         if (str[0] == '\0' || strnlen(str, DLADM_WLAN_MAX_ESSID_LEN + 2) ==
 882             DLADM_WLAN_MAX_ESSID_LEN + 2)
 851         if (str[0] == '\0' || strlen(str) > DLADM_WLAN_MAX_ESSID_LEN - 1)
 883                 return (DLADM_STATUS_BADARG);

 885         essid->we_length=strnlen(str, DLADM_WLAN_MAX_ESSID_LEN + 1);
 886         (void) memcpy(essid->we_bytes, str, essid->we_length);
 854         (void) strlcpy(essid->we_bytes, str, DLADM_WLAN_MAX_ESSID_LEN);
 887         return (DLADM_STATUS_OK);
 888 }
_____unchanged_portion_omitted_

 991 /* legacy ioctl for WL_SCAN and WL_DISASSOCIATE */

 993 #endif /* ! codereview */
 994 dladm_status_t
 995 i_dladm_wlan_legacy_ioctl(dladm_handle_t handle, datalink_id_t linkid,
 996     wldp_t *gbuf, uint_t id, size_t len, uint_t cmd, size_t cmdlen)
 997 {
 998         char                    linkname[MAXPATHLEN];
 999         int                     fd, rc;
1000         struct  strioctl        stri;
1001         uint32_t                flags;
1002         dladm_status_t          status;
1003         uint32_t                media;
1004         char                    link[MAXLINKNAMELEN];

1006         if ((status = dladm_datalink_id2info(handle, linkid, &flags, NULL,
1007             &media, link, MAXLINKNAMELEN)) != DLADM_STATUS_OK) {
1008                 return (status);
1009         }

1011         if (media != DL_WIFI)
1012                 return (DLADM_STATUS_BADARG);

1014         if (!(flags & DLADM_OPT_ACTIVE))
1015                 return (DLADM_STATUS_TEMPONLY);

1017         /*
1018          * dlpi_open() is not used here because libdlpi depends on libdladm,
1019          * and we do not want to introduce recursive dependencies.
1020          */
1021         (void) snprintf(linkname, MAXPATHLEN, "/dev/net/%s", link);
1022         if ((fd = open(linkname, O_RDWR)) < 0)
1023                 return (dladm_errno2status(errno));

1025         gbuf->wldp_type = NET_802_11;
1026         gbuf->wldp_id   = id;
```



```
1027         gbuf->wldp_length = len;

1029         stri.ic_timout  = 0;
1030         stri.ic_dp      = (char *)gbuf;
1031         stri.ic_cmd     = cmd;
1032         stri.ic_len     = cmdlen;

1034         if ((rc = ioctl(fd, I_STR, &stri)) != 0) {
1035                 if (rc > 0) {
1036                         /*
1037                          * Non-negative return value indicates the specific
1038                          * operation failed and the reason for the failure
1039                          * was stored in gbuf->wldp_result.
1040                          */
1041                         status = dladm_wlan_wlresult2status(gbuf);
1042                 } else {
1043                         /*
1044                          * Negative return value indicates the ioctl failed.
1045                          */
1046                         status = dladm_errno2status(errno);
1047                 }
1048         }
1049         (void) close(fd);
1050         return (status);
1051 }

1053 static dladm_status_t
1054 do_cmd_ioctl(dladm_handle_t handle, datalink_id_t linkid, void *buf,
1055     int buflen, uint_t cmd)
1056 {
1057         wldp_t *gbuf;
1058         dladm_status_t status = DLADM_STATUS_OK;

1060         if ((gbuf = malloc(MAX_BUF_LEN)) == NULL)
1061                 return (DLADM_STATUS_NOMEM);

1063         (void) memset(gbuf, 0, MAX_BUF_LEN);
1064         status = i_dladm_wlan_legacy_ioctl(handle, linkid, gbuf, cmd,
1065             WLDP_BUFSIZE, WLAN_COMMAND, sizeof (wldp_t));
1066         (void) memcpy(buf, gbuf->wldp_buf, buflen);
1067         free(gbuf);
1068         return (status);
1069 }

1071 static dladm_status_t
1072 do_scan(dladm_handle_t handle, datalink_id_t linkid, void *buf, int buflen)
1073 {
1074         return (do_cmd_ioctl(handle, linkid, buf, buflen, WL_SCAN));
1075 }

1077 static dladm_status_t
1078 do_disconnect(dladm_handle_t handle, datalink_id_t linkid, void *buf,
1079     int buflen)
1080 {
1081         char ifname[MAXLINKNAMELEN];
1082         if ((dladm_datalink_id2info(handle, linkid, NULL, NULL, NULL, ifname,
1083             sizeof (ifname))) != DLADM_STATUS_OK)
1084                         return DLADM_STATUS_FAILED;
1085                 (void) wpa_instance_delete(ifname);
 959         if (do_get_wpamode(handle, linkid, buf, buflen) == 0 &&
 960             ((wl_wpa_t *)(buf))->wpa_flag > 0)
 961                 (void) wpa_instance_delete(handle, linkid);

1087         return (do_cmd_ioctl(handle, linkid, buf, buflen, WL_DISASSOCIATE));
1088 }
```

```
/*
 * 'net80211' ioctls wrappers
 * wlan properties are not currenlty merged into the Brussels framework.
 * we should implement a common interface like this:http://pastebin.com/B05BkQJ2
 */

dladm_status_t
dladm_wlan_get_esslist(dladm_handle_t handle, datalink_id_t linkid, void *buf,
static dladm_status_t
do_get_esslist(dladm_handle_t handle, datalink_id_t linkid, void *buf,
    int buflen)
{
        return (i_dladm_wlan_param(handle, linkid, buf, MAC_PROP_WL_ESS_LIST,
            buflen, B_FALSE));
}

dladm_status_t
dladm_wlan_get_bssid(dladm_handle_t handle, datalink_id_t linkid, void *buf, int
static dladm_status_t
do_get_bssid(dladm_handle_t handle, datalink_id_t linkid, void *buf, int buflen)
{
        return (i_dladm_wlan_param(handle, linkid, buf, MAC_PROP_WL_BSSID,
            buflen, B_FALSE));
}

dladm_status_t
dladm_wlan_get_essid(dladm_handle_t handle, datalink_id_t linkid, void *buf, int
static dladm_status_t
do_get_essid(dladm_handle_t handle, datalink_id_t linkid, void *buf, int buflen)
{
        return (i_dladm_wlan_param(handle, linkid, buf, MAC_PROP_WL_ESSID,
            buflen, B_FALSE));
}
______unchanged_portion_omitted_

dladm_status_t
dladm_wlan_set_bsstype(dladm_handle_t handle, datalink_id_t linkid,
static dladm_status_t
do_set_bsstype(dladm_handle_t handle, datalink_id_t linkid,
    dladm_wlan_bsstype_t *bsstype)
{
        wl_bss_type_t   ibsstype;

        switch (*bsstype) {
        case DLADM_WLAN_BSSTYPE_BSS:
                ibsstype = WL_BSS_BSS;
                break;
        case DLADM_WLAN_BSSTYPE_IBSS:
                ibsstype = WL_BSS_IBSS;
                break;
        default:
                ibsstype = WL_BSS_BSS;
                ibsstype = WL_BSS_ANY;
                break;
        }
        return (i_dladm_wlan_param(handle, linkid, &ibsstype,
            MAC_PROP_WL_BSSTYPE, sizeof (ibsstype), B_TRUE));
}

dladm_status_t
dladm_wlan_set_authmode(dladm_handle_t handle, datalink_id_t linkid,
static dladm_status_t
do_set_authmode(dladm_handle_t handle, datalink_id_t linkid,
    dladm_wlan_auth_t *auth)
{
        wl_authmode_t   auth_mode;
```



```
        switch (*auth) {
        case DLADM_WLAN_AUTH_NONE:
                return (DLADM_STATUS_OK);
#endif /* ! codereview */
        case DLADM_WLAN_AUTH_OPEN:
                auth_mode = WL_OPENSYSTEM;
                break;
        case DLADM_WLAN_AUTH_SHARED:
                auth_mode = WL_SHAREDKEY;
                break;
        default:
                return (DLADM_STATUS_NOTSUP);
        }
        return (i_dladm_wlan_param(handle, linkid, &auth_mode,
            MAC_PROP_WL_AUTH_MODE, sizeof (auth_mode), B_TRUE));
}

/* should be renamed to a more appropriate name since it is only for WEP !!*/
dladm_status_t
dladm_wlan_set_encryption(dladm_handle_t handle, datalink_id_t linkid,
static dladm_status_t
do_set_encryption(dladm_handle_t handle, datalink_id_t linkid,
    dladm_wlan_secmode_t *secmode)
{
        wl_encryption_t encryption;

        switch (*secmode) {
        case DLADM_WLAN_SECMODE_NONE:
                encryption = WL_NOENCRYPTION;
                break;
        case DLADM_WLAN_SECMODE_WEP:
                encryption = WL_ENC_WEP;
                break;
        case DLADM_WLAN_SECMODE_PSK:
        case DLADM_WLAN_SECMODE_WPA:
                return (0);
        default:
                return (DLADM_STATUS_NOTSUP);
        }
        return (i_dladm_wlan_param(handle, linkid, &encryption,
            MAC_PROP_WL_ENCRYPTION, sizeof (encryption), B_TRUE));
}
______unchanged_portion_omitted_

dladm_status_t
dladm_wlan_set_essid(dladm_handle_t handle, datalink_id_t linkid,
    const uint8_t *essid, size_t essid_len)
static dladm_status_t
do_set_essid(dladm_handle_t handle, datalink_id_t linkid,
    dladm_wlan_essid_t *essid)
{
        wl_essid_t      iessid;

        (void) memset(&iessid, 0, essid_len);
        (void) memset(&iessid, 0, sizeof (essid));

        if (essid != NULL && essid_len != 0) {
                iessid.wl_essid_length = essid_len;
                (void) memcpy(iessid.wl_essid_essid, essid,
                    iessid.wl_essid_length);
        if (essid != NULL && essid->we_bytes[0] != '\0') {
                iessid.wl_essid_length = strlen(essid->we_bytes);
                (void) strlcpy(iessid.wl_essid_essid, essid->we_bytes,
                    sizeof (iessid.wl_essid_essid));
        } else {
```

```
1286                 return (DLADM_STATUS_BADARG);
1287         }
1288         return (i_dladm_wlan_param(handle, linkid, &iessid, MAC_PROP_WL_ESSID,
1289             sizeof (iessid), B_TRUE));
1290 }

1292 dladm_status_t
1293 do_set_bssid(dladm_handle_t handle, datalink_id_t linkid,
1294     dladm_wlan_bssid_t *bssid)
1295 {
1296         wl_bssid_t      ibssid;

1298         if (bssid != NULL && bssid->wb_bytes != 0) {
1299                 (void) memcpy(ibssid, bssid->wb_bytes, DLADM_WLAN_BSSID_LEN);
1300         } else {
1301                 return (DLADM_STATUS_BADARG);
1302         }
1303         return (i_dladm_wlan_param(handle, linkid, &ibssid, MAC_PROP_WL_BSSID,
1304             sizeof (ibssid), B_TRUE));
1305 }

1307 dladm_status_t
1308 dladm_wlan_set_channel(dladm_handle_t handle, datalink_id_t linkid,
1145 static dladm_status_t
1146 do_set_channel(dladm_handle_t handle, datalink_id_t linkid,
1309     dladm_wlan_channel_t *channel)
1310 {
1311         wl_phy_conf_t phy_conf;

1313         if (*channel > MAX_CHANNEL_NUM)
1314                 return (DLADM_STATUS_BADVAL);

1316         (void) memset(&phy_conf, 0xff, sizeof (phy_conf));
1317         phy_conf.wl_phy_dsss_conf.wl_dsss_channel = *channel;

1319         return (i_dladm_wlan_param(handle, linkid, &phy_conf,
1320             MAC_PROP_WL_PHY_CONFIG, sizeof (phy_conf), B_TRUE));
1321 }

1323 dladm_status_t
1161 static dladm_status_t
1324 do_set_createibss(dladm_handle_t handle, datalink_id_t linkid,
1325     boolean_t *create_ibss)
1326 {
1327         wl_create_ibss_t cr = (wl_create_ibss_t)(*create_ibss);

1329         return (i_dladm_wlan_param(handle, linkid, &cr, MAC_PROP_WL_CREATE_IBSS,
1330             sizeof (cr), B_TRUE));
1331 }

1333 static void
1334 generate_essid(dladm_wlan_essid_t *essid)
1335 {
1336         char ssid_temp[DLADM_WLAN_MAX_ESSID_LEN];
1337 #endif /* ! codereview */
1338         srandom(gethrtime());
1339         snprintf(ssid_temp, DLADM_WLAN_MAX_ESSID_LEN, "Illumos-%s", random());
1340         (void) memcpy(essid->we_bytes,ssid_temp,16);
1341         essid->we_length=16;
1174         (void) snprintf(essid->we_bytes, DLADM_WLAN_MAX_ESSID_LEN, "%d",
1175             random());
1342 }

1344 dladm_status_t
1178 static dladm_status_t
1345 do_get_capability(dladm_handle_t handle, datalink_id_t linkid, void *buf,
```



```
1346     int buflen)
1347 {
1348         return (i_dladm_wlan_param(handle, linkid, buf, MAC_PROP_WL_CAPABILITY,
1349             buflen, B_FALSE));
1350 }
______unchanged_portion_omitted_

1360 /* WPA support routines */
1194 dladm_status_t
1195 dladm_wlan_wpa_get_sr(dladm_handle_t handle, datalink_id_t linkid,
1196     dladm_wlan_ess_t *sr, uint_t escnt, uint_t *estot)
1197 {
1198         int             i, n;
1199         wl_wpa_ess_t    *es;
1200         dladm_status_t  status;

1202         es = malloc(WLDP_BUFSIZE);
1203         if (es == NULL)
1204                 return (DLADM_STATUS_NOMEM);

1206         status = i_dladm_wlan_param(handle, linkid, es, MAC_PROP_WL_SCANRESULTS,
1207             WLDP_BUFSIZE, B_FALSE);

1209         if (status == DLADM_STATUS_OK) {
1210                 n = (es->count > escnt) ? escnt : es->count;
1211                 for (i = 0; i < n; i ++) {
1212                         (void) memcpy(sr[i].we_bssid.wb_bytes, es->ess[i].bssid,
1213                             DLADM_WLAN_BSSID_LEN);
1214                         sr[i].we_ssid_len = es->ess[i].ssid_len;
1215                         (void) memcpy(sr[i].we_ssid.we_bytes, es->ess[i].ssid,
1216                             es->ess[i].ssid_len);
1217                         sr[i].we_wpa_ie_len = es->ess[i].wpa_ie_len;
1218                         (void) memcpy(sr[i].we_wpa_ie, es->ess[i].wpa_ie,
1219                             es->ess[i].wpa_ie_len);
1220                         sr[i].we_freq = es->ess[i].freq;
1221                 }
1222                 *estot = n;
1223         }

1225         free(es);
1226         return (status);
1227 }

1362 dladm_status_t
1363 dladm_wlan_wpa_set_ie(dladm_handle_t handle, datalink_id_t linkid,
1364     const uint8_t *wpa_ie, size_t wpa_ie_len)
1231     uint8_t *wpa_ie, uint_t wpa_ie_len)
1365 {
1366         wl_wpa_ie_t *ie;
1367         uint_t len;
1368         dladm_status_t  status;

1370         if (wpa_ie_len > DLADM_WLAN_MAX_WPA_IE_LEN)
1371                 return (DLADM_STATUS_BADARG);
1372         len = sizeof (wl_wpa_ie_t) + wpa_ie_len;
1373         ie = malloc(len);
1374         if (ie == NULL)
1375                 return (DLADM_STATUS_NOMEM);

1377         (void) memset(ie, 0, len);
1378         ie->wpa_ie_len = wpa_ie_len;
1379         (void) memcpy(ie->wpa_ie, wpa_ie, wpa_ie_len);

1381         status = i_dladm_wlan_param(handle, linkid, ie, MAC_PROP_WL_SETOPTIE,
1382             len, B_TRUE);
1383         free(ie);
```

```
1385         return (status);
1386 }
_____unchanged_portion_omitted_

1461 dladm_status_t
1462 dladm_wlan_wpa_set_mlme(dladm_handle_t handle, datalink_id_t linkid,
1463     dladm_wlan_mlme_op_t op, dladm_wlan_reason_t reason, const uint8_t *bssid)
1330     dladm_wlan_mlme_op_t op, dladm_wlan_reason_t reason,
1331     dladm_wlan_bssid_t *bssid)
1464 {
1465         wl_mlme_t mlme;

1467         (void) memset(&mlme, 0, sizeof (wl_mlme_t));
1468         switch (op) {
1469         case DLADM_WLAN_MLME_ASSOC:
1470                 mlme.im_op = IEEE80211_MLME_ASSOC;
1471                 break;
1472         case DLADM_WLAN_MLME_DISASSOC:
1473                 mlme.im_op = IEEE80211_MLME_DISASSOC;
1474                 break;
1475         default:
1476                 return (DLADM_STATUS_BADARG);
1477         }
1478         mlme.im_reason = reason;
1479         if (bssid != NULL)
1480             memcpy(mlme.im_macaddr, bssid, DLADM_WLAN_BSSID_LEN);
1348             (void) memcpy(mlme.im_macaddr, bssid->wb_bytes,
1349                 DLADM_WLAN_BSSID_LEN);

1482         return (i_dladm_wlan_param(handle, linkid, &mlme, MAC_PROP_WL_MLME,
1483             sizeof (mlme), B_TRUE));
1484 }

1486  /*
1487   * This routine is used for opening a control interface to wpa_supplicant
1488   * global interface ctrl_path, /var/run/wpa_supplicant-global. This path
1489   * is configured in network/wpa_supplicant service manifest.
1490   * network/wpa_supplicant service must be running.
1355 /*
1356  * routines of create instance
1491   */
1492 static int
1493 wpa_instance_create(char *ifname)
1358 static scf_propertygroup_t *
1359 add_property_group_to_instance(scf_handle_t *handle, scf_instance_t *instance,
1360     const char *pg_name, const char *pg_type)
1494 {
1495         struct wpa_ctrl *ctrl_global;
1496         const char *global_path = CTRL_IFACE_GLOBAL;

1498         /*
1499          * When we'll have a configuration backend that supports know_wlans.conf
1500          * we could add this file direclty in the third parameter
1501          */
1502         char *interface_add[] = {"interface_add", ifname, """", "solaris",
1503             CTRL_IFACE_DIR};
1362         scf_propertygroup_t *pg;

1505         if (ifname == NULL)
1506                 return -1;
1364         pg = scf_pg_create(handle);
1365         if (pg == NULL)
1366                 return (NULL);

1508         ctrl_global = wpa_ctrl_open(global_path);
```



```
1509         if (ctrl_global == NULL) {
1510                 return -1;
1368         if (scf_instance_add_pg(instance, pg_name, pg_type, 0, pg) != 0) {
1369                 scf_pg_destroy(pg);
1370                 return (NULL);
1511         }

1513         if (wpa_request(ctrl_global, 5, interface_add))
1514                 return -1;
1373         return (pg);
1374 }

1376 static dladm_status_t
1377 add_new_property(scf_handle_t *handle, const char *prop_name,
1378     scf_type_t type, const char *val, scf_transaction_t *tx)
1379 {
1380         scf_value_t *value = NULL;
1381         scf_transaction_entry_t *entry = NULL;

1516         wpa_ctrl_close(ctrl_global);
1383         entry = scf_entry_create(handle);
1384         if (entry == NULL)
1385                 goto out;

1518         /* wpa_s now executes wpa_driver_solaris_init */
1387         value = scf_value_create(handle);
1388         if (value == NULL)
1389                 goto out;

1520         return 0;
1391         if (scf_transaction_property_new(tx, entry, prop_name, type) != 0)
1392                 goto out;

1394         if (scf_value_set_from_string(value, type, val) != 0)
1395                 goto out;

1397         if (scf_entry_add_value(entry, value) != 0)
1398                 goto out;

1400         return (DLADM_STATUS_OK);

1402 out:
1403         if (value != NULL)
1404                 scf_value_destroy(value);
1405         if (entry != NULL)
1406                 scf_entry_destroy(entry);

1408         return (DLADM_STATUS_FAILED);
1521 }

1523 static int
1524 wpa_instance_delete(char *ifname)
1411 static dladm_status_t
1412 add_pg_method(scf_handle_t *handle, scf_instance_t *instance,
1413     const char *pg_name, const char *flags)
1525 {
1526         /*struct wpa_ctrl *ctrl_conn;*/
1527         struct wpa_ctrl *ctrl_global;
1415         int                     rv, size;
1416         dladm_status_t          status = DLADM_STATUS_FAILED;
1417         char                    *command = NULL;
1418         scf_transaction_t       *tran = NULL;
1419         scf_propertygroup_t     *pg;

1421         pg = add_property_group_to_instance(handle, instance,
1422             pg_name, SCF_GROUP_METHOD);
```

```
1423         if (pg == NULL)
1424                 goto out;

1426         tran = scf_transaction_create(handle);
1427         if (tran == NULL)
1428                 goto out;

1430         size = strlen(SVC_METHOD) + strlen("  ") + strlen(flags) + 1;
1431         command = malloc(size);
1432         if (command == NULL) {
1433                 status = DLADM_STATUS_NOMEM;
1434                 goto out;
1435         }
1436         (void) snprintf(command, size, "%s %s", SVC_METHOD, flags);

1529         const char *global_path = CTRL_IFACE_GLOBAL;
1530         /*char *cfile = NULL;*/
1531         char *interface_remove[] = {"interface_remove", ifname};
1532         /* char *cmd_disconnect[] = {"disconnect"}; */
1533         /* int flen, res;*/
1438         do {
1439                 if (scf_transaction_start(tran, pg) != 0)
1440                         goto out;

1535         if (ifname == NULL)
1536                 return -1;
1442                 if (add_new_property(handle, SCF_PROPERTY_EXEC,
1443                     SCF_TYPE_ASTRING, command, tran) != DLADM_STATUS_OK) {
1444                         goto out;
1445                 }

1538 /*
1447                 rv = scf_transaction_commit(tran);
1448                 switch (rv) {
1449                 case 1:
1450                         status = DLADM_STATUS_OK;
1451                         goto out;
1452                 case 0:
1453                         scf_transaction_destroy_children(tran);
1454                         if (scf_pg_update(pg) == -1) {
1455                                 goto out;
1456                         }
1457                         break;
1458                 case -1:
1459                 default:
1460                         goto out;
1461                 }
1462         } while (rv == 0);

1464 out:
1465         if (tran != NULL) {
1466                 scf_transaction_destroy_children(tran);
1467                 scf_transaction_destroy(tran);
1468         }

1540          * Get the instance name of the existing control interface.
1541          * Control interface will use the link name as the UNIX socket filename
1542          * in CTRL_IFACE_DIR
1470         if (pg != NULL)
1471                 scf_pg_destroy(pg);

1544         if (cfile == NULL) {
1545                 flen = strlen(CTRL_IFACE_DIR) + strlen(ifname) + 2;
1546                 cfile = malloc(flen);
1547                 if (cfile == NULL)
1548                         return -1;
```



```
1549                 res = snprintf(cfile, flen, "%s/%s", CTRL_IFACE_DIR, ifname);
1550                 if (res < 0 || res >= flen) {
1551                         free(cfile);
1552                         return -1;
1473         if (command != NULL)
1474                 free(command);

1476         return (status);
1477 }

1479 static dladm_status_t
1480 do_create_instance(scf_handle_t *handle, scf_service_t *svc,
1481     const char *instance_name, const char *command)
1482 {
1483         dladm_status_t status = DLADM_STATUS_FAILED;
1484         char *buf;
1485         ssize_t max_fmri_len;
1486         scf_instance_t *instance;

1488         instance = scf_instance_create(handle);
1489         if (instance == NULL)
1490                 goto out;

1492         if (scf_service_add_instance(svc, instance_name, instance) != 0) {
1493                 if (scf_error() == SCF_ERROR_EXISTS)
1494                         /* Let the caller deal with the duplicate instance */
1495                         status = DLADM_STATUS_EXIST;
1496                 goto out;
1497         }

1499         if (add_pg_method(handle, instance, "start",
1500             command) != DLADM_STATUS_OK) {
1501                 goto out;
1502         }

1504         /* enabling the instance */
1505         max_fmri_len = scf_limit(SCF_LIMIT_MAX_FMRI_LENGTH);
1506         if ((buf = malloc(max_fmri_len + 1)) == NULL)
1507                 goto out;

1509         if (scf_instance_to_fmri(instance, buf, max_fmri_len + 1) > 0) {
1510                 if ((smf_disable_instance(buf, 0) != 0) ||
1511                     (smf_enable_instance(buf, SMF_TEMPORARY) != 0)) {
1512                         goto out;
1553                 }
1514                 status = DLADM_STATUS_OK;
1554         }

1556         ctrl_conn = wpa_ctrl_open(cfile);
1517 out:
1518         if (instance != NULL)
1519                 scf_instance_destroy(instance);
1520         return (status);
1521 }

1558         free(cfile);
1523 static dladm_status_t
1524 create_instance(const char *instance_name, const char *command)
1525 {
1526         dladm_status_t status = DLADM_STATUS_FAILED;
1527         scf_service_t *svc = NULL;
1528         scf_handle_t *handle = NULL;

1560         if (ctrl_conn == NULL)
1561                 return -1;
1530         handle = scf_handle_create(SCF_VERSION);
```

```
1531         if (handle == NULL)
1532                 goto out;

1563         if (wpa_request(ctrl_conn, 1, cmd_disconnect))
1564                 return -1;
1534         if (scf_handle_bind(handle) == -1)
1535                 goto out;

1566         wpa_ctrl_close(ctrl_conn);
1567 */
1537         if ((svc = scf_service_create(handle)) == NULL)
1538                 goto out;

1569         /* remove interface */
1570         ctrl_global = wpa_ctrl_open(global_path);
1571         if (ctrl_global == NULL) {
1572                 return -1;
1573         }
1540         if (scf_handle_decode_fmri(handle, SERVICE_NAME, NULL, svc,
1541             NULL, NULL, NULL, SCF_DECODE_FMRI_EXACT) != 0)
1542                 goto out;

1544         status = do_create_instance(handle, svc, instance_name, command);

1575         if (wpa_request(ctrl_global, 2, interface_remove))
1576                 return -1;
1546 out:
1547         if (svc != NULL)
1548                 scf_service_destroy(svc);

1578         wpa_ctrl_close(ctrl_global);
1550         if (handle != NULL) {
1551                 (void) scf_handle_unbind(handle);
1552                 scf_handle_destroy(handle);
1553         }

1580         return 0;
1555         return (status);
1581 }

1583 #define MAC2STR(a) (a)[0], (a)[1], (a)[2], (a)[3], (a)[4], (a)[5]
1584 #define MACSTR "%02x:%02x:%02x:%02x:%02x:%02x"
1558 /*
1559  * routines of delete instance
1560  */
1561 #define DEFAULT_TIMEOUT 60000000
1562 #define INIT_WAIT_USECS 50000

1586 static int wpa_network_config(char *ifname, dladm_wlan_attr_t *attrp,
1587         dladm_wlan_key_t *keys, boolean_t *create_ibss, char *identity)
1564 static void
1565 wait_until_disabled(scf_handle_t *handle, char *fmri)
1566 {
1567         char            *state;
1568         useconds_t      max;
1569         useconds_t      usecs;
1570         uint64_t        *cp = NULL;
1571         scf_simple_prop_t *sp = NULL;

1573         max = DEFAULT_TIMEOUT;

1575         if (((sp = scf_simple_prop_get(handle, fmri, "stop",
1576             SCF_PROPERTY_TIMEOUT)) != NULL) &&
1577             ((cp = scf_simple_prop_next_count(sp)) != NULL) && (*cp != 0))
1578                 max = (*cp) * 1000000;  /* convert to usecs */
```



```
1580         if (sp != NULL)
1581                 scf_simple_prop_free(sp);

1583         for (usecs = INIT_WAIT_USECS; max > 0; max -= usecs) {
1584                 /* incremental wait */
1585                 usecs *= 2;
1586                 usecs = (usecs > max) ? max : usecs;

1588                 (void) usleep(usecs);

1590                 /* Check state after the wait */
1591                 if ((state = smf_get_state(fmri)) != NULL) {
1592                         if (strcmp(state, "disabled") == 0)
1593                                 return;
1594                 }
1595         }
1596 }

1598 static dladm_status_t
1599 delete_instance(const char *instance_name)
1588 {
1589         struct wpa_ctrl *ctrl_conn;
1601         dladm_status_t  status = DLADM_STATUS_FAILED;
1602         char            *buf;
1603         ssize_t         max_fmri_len;
1604         scf_scope_t     *scope = NULL;
1605         scf_service_t   *svc = NULL;
1606         scf_handle_t    *handle = NULL;
1607         scf_instance_t  *instance;

1591         char *cfile = NULL;
1592         int i, flen, res;
1609         handle = scf_handle_create(SCF_VERSION);
1610         if (handle == NULL)
1611                 goto out;

1594         char *ap_scan[] = {"ap_scan", "1"};
1595         char *add_network[] = {"add_network"};
1596         char *enable_network[] = {"enable_network", "0"};
1597         char *set_network[] = {"set_network", "0", NULL, NULL };
1598         char *network_props[] = { "ssid", "key_mgmt", "pairwise", "group",
1599             "auth_alg", "proto"};
1600         char *network_vals[6];
1601         char *hexascii = NULL;
1602         char ap_bssid[DLADM_WLAN_BSSID_LEN*3];
1603         char *cmd_bssid[] = { "bssid", "0", ap_bssid };
1613         if (scf_handle_bind(handle) == -1)
1614                 goto out;

1605         if (ifname == NULL)
1606                 return -1;
1616         if ((scope = scf_scope_create(handle)) == NULL)
1617                 goto out;

1608         for (i=0; i<6; i++)
1609                 network_vals[1] = NULL;
1619         if ((svc = scf_service_create(handle)) == NULL)
1620                 goto out;

1622         if (scf_handle_get_scope(handle, SCF_SCOPE_LOCAL, scope) == -1)
1623                 goto out;

1625         if (scf_scope_get_service(scope, SERVICE_NAME, svc) < 0)
1626                 goto out;

1628         instance = scf_instance_create(handle);
```

```
1629         if (instance == NULL)
1630                 goto out;

1632         if (scf_service_get_instance(svc, instance_name, instance) != 0) {
1633                 scf_error_t scf_errnum = scf_error();

1635                 if (scf_errnum == SCF_ERROR_NOT_FOUND)
1636                         status = DLADM_STATUS_OK;

1638                 scf_instance_destroy(instance);
1639                 goto out;
1640         }

1642         max_fmri_len = scf_limit(SCF_LIMIT_MAX_FMRI_LENGTH);
1643         if ((buf = malloc(max_fmri_len + 1)) == NULL) {
1644                 scf_instance_destroy(instance);
1645                 goto out;
1646         }

1648         if (scf_instance_to_fmri(instance, buf, max_fmri_len + 1) > 0) {
1649                 char *state;

1651                 state = smf_get_state(buf);
1652                 if (state && (strcmp(state, SCF_STATE_STRING_ONLINE) == 0 ||
1653                     strcmp(state, SCF_STATE_STRING_DEGRADED) == 0)) {
1654                         if (smf_disable_instance(buf, 0) == 0) {
1611         /*
1612          * Control interface will use the link name as the UNIX socket filename
1613          * in CTRL_IFACE_DIR
1656                                  * Wait for some time till timeout to avoid
1657                                  * a race with scf_instance_delete() below.
1614          */
1615         if (cfile == NULL) {
1616                 flen = strlen(CTRL_IFACE_DIR) + strlen(ifname) + 2;
1617                 cfile = malloc(flen);
1618                 if (cfile == NULL)
1619                         return -1;
1620                 res = snprintf(cfile, flen, "%s/%s", CTRL_IFACE_DIR, ifname);
1621                 if (res < 0 || res >= flen) {
1622                         free(cfile);
1623                         return -1;
1659                                 wait_until_disabled(handle, buf);
1660                         }
1624                 }
1625         }

1627         ctrl_conn = wpa_ctrl_open(cfile);
1628         free(cfile);

1630         if (ctrl_conn == NULL)
1631                 return -1;

1633         /* set ap_scan=2 */
1634         if (wpa_request(ctrl_conn, 2, ap_scan))
1635                 return -1;

1637         /* add empty wifi network block */
1638         (void) wpa_request(ctrl_conn, 1, add_network);

1640         /* leave bssid out if strict_bssid=false?*/
1641         snprintf(ap_bssid, 3 * DLADM_WLAN_BSSID_LEN, MACSTR,
1642             MAC2STR(attrp->wa_bssid.wb_bytes));
1643         (void) wpa_request(ctrl_conn, 3, cmd_bssid);

1645         /* ssid string needs to be quoted */
1646         network_vals[0] = malloc(attrp->wa_essid.we_length + 3);
```



```
1647         snprintf(network_vals[0], attrp->wa_essid.we_length + 3, "\"%s\"",
1648             attrp->wa_essid.we_bytes);
1649         /* we will add support for static WEP config. with wpa_s later */
1650         network_vals[4] = (char*)dladm_wlan_auth2str(&attrp->wa_auth, malloc(5))
1664         if (scf_instance_delete(instance) != 0) {
1665                 scf_instance_destroy(instance);
1666                 goto out;
1667         }

1652         /*
1653         if (*create_ibss) {
1654                 network_vals[1]="WPA-NONE";
1655                 network_vals[2]="NONE";
1656                 network_vals[3]="TKIP";
1657                 network_vals[5]="WPA";
1658                  * mode=1
1659                  * frequency=2412
1660                  * psk="secret passphrase"
1661                  * ssid="test adhoc"
1662         } else {
1663                 network_vals[1] = wpa_key_mgmt_txt(attrp->wa_ie.key_mgmt);
1664                 network_vals[2] = wpa_cipher_txt(attrp->wa_ie.pairwise_cipher);
1665                 network_vals[3] = wpa_cipher_txt(attrp->wa_ie.group_cipher);
1666                 network_vals[5] = ((attrp->wa_ie.proto == 2) ? "RSN" : "WPA");
1667         }

1669         for (i=0; i<6; i++) {
1670                 set_network[2] = network_props[i];
1671                 set_network[3] = network_vals[i];
1672                 (void) wpa_request(ctrl_conn, 4, set_network);
1673         } */

1675         set_network[2] = network_props[0];
1676         set_network[3] = network_vals[0];
1677         wpa_request(ctrl_conn, 4, set_network);
1678         free(network_vals[0]);
1679         free(network_vals[4]);
1669         scf_instance_destroy(instance);

1671         status = DLADM_STATUS_OK;

1673 out:
1674         if (svc != NULL)
1675                 scf_service_destroy(svc);

1677         if (scope != NULL)
1678                 scf_scope_destroy(scope);

1680         if (handle != NULL) {
1681                 (void) scf_handle_unbind(handle);
1682                 scf_handle_destroy(handle);
1683         }

1685         return (status);
1686 }

1688 static dladm_status_t
1689 wpa_instance_create(dladm_handle_t handle, datalink_id_t linkid, void *key)
1690 {
1691         dladm_status_t  status = DLADM_STATUS_FAILED;
1692         char            *command = NULL;
1693         char            *wk_name = ((dladm_wlan_key_t *)key)->wk_name;
1694         int             size;
1695         char            instance_name[MAXLINKNAMELEN];

1681         /*
```

```
1682          * If we use hexascii strings, wpa_s does not require quotation, and
1683          * we would be able to pass it directly to wpa_request.
1684          * Currently getsecobj returns the string value, not hexascii
1698          * Use the link name as the instance name of the network/wpad service.
1685          */
1686         hexascii = malloc(keys->wk_len + 3);
1687         snprintf(hexascii, keys->wk_len + 3, "\"%s\"", keys->wk_val);
1700         status = dladm_datalink_id2info(handle, linkid, NULL, NULL, NULL,
1701             instance_name, sizeof (instance_name));
1702         if (status != DLADM_STATUS_OK)
1703                 goto out;

1689         if (keys->wk_class == DLADM_SECOBJ_CLASS_PSK) {
1690                 set_network[2] = "psk";
1691                 set_network[3] = hexascii;
1692                 (void) wpa_request(ctrl_conn, 4, set_network);
1693         } else {
1694                 char *cmd_eap[] = {"set_network", "0", "eap", NULL};
1695                 set_network[2] = "identity";
1696                 set_network[3] = identity;
1697                 (void) wpa_request(ctrl_conn, 4, set_network);
1698                 set_network[2] = "ca_cert";
1699                 set_network[3] = "\"/etc/cert/ca.pem\"";
1700                 (void) wpa_request(ctrl_conn, 4, set_network);
1701                  */
1702                 if (keys->wk_class == DLADM_SECOBJ_CLASS_TLS) {
1703                         char *tls_props[] = {"engine", "engine_id", "key_id",
1704                             "pin", "client_cert"};
1705                         char *tls_vals[] = {"1", "pkcs11", "TODO", hexascii,
1706                             "DEFAULT_PATH"};
1707                         cmd_eap[3] = "TLS";
1708                         (void) wpa_request(ctrl_conn, 4, cmd_eap);
1709                         for (i=0; i<6; i++) {
1710                                 char *cmd_tls[] = {"set_network", "0",
1711                                     tls_props[i], tls_vals[i]};
1712                                 (void) wpa_request(ctrl_conn, 4, cmd_tls);
1705         size = strlen(instance_name) + strlen(" -i  -k ") + strlen(wk_name) + 1;
1706         command = malloc(size);
1707         if (command == NULL) {
1708                 status = DLADM_STATUS_NOMEM;
1709                 goto out;
1713                 }
1714                 } else {
1715                         set_network[2] = "password";
1716                         set_network[3] = hexascii;
1717                         (void) wpa_request(ctrl_conn, 4, set_network);
1718                         if (keys->wk_class == DLADM_SECOBJ_CLASS_TTLS) {
1719                                 cmd_eap[3] = "TTLS";
1720                                 (void) wpa_request(ctrl_conn, 4, cmd_eap);
1721                                 /* optional
1722                                 set_network[2] = "anonymous_identity";
1723                                 set_network[3] = "DEFAULT_ANONYMOUS";
1724                                 (void) wpa_request(ctrl_conn, 4, set_network);*/
1725                                 set_network[2] = "phase2";
1726                                 set_network[3] = "\"auth=PAP\"";
1711         (void) snprintf(command, size, "-i %s -k %s", instance_name, wk_name);

1713         status = create_instance(instance_name, command);
1714         if (status == DLADM_STATUS_EXIST) {
1727                                 /*
1728                                  * other possible values are: auth=MSCHAPV2 auth
1729                                  * we do not support autheap=TLS for eap-ttls
1716                  * Delete the existing instance and create a new instance
1717                  * with the supplied arguments.
1730                                  */
1731                                 (void) wpa_request(ctrl_conn, 4, set_network);
```



```
1732                         } else {
1733                                 cmd_eap[3] = "PEAP";
1734                                 (void) wpa_request(ctrl_conn, 4, cmd_eap);
1735                                 set_network[2] = "phase2";
1736                                 set_network[3] = "\"auth=MSCHAPV2\"";
1737                                 (void) wpa_request(ctrl_conn, 4, set_network);
1719                 if ((status = delete_instance(instance_name)) ==
1720                     DLADM_STATUS_OK) {
1721                         status = create_instance(instance_name, command);
1738                         }
1739                 }
1740         }
1741         (void) wpa_request(ctrl_conn, 2, enable_network);
1742         /*
1743          * the wpa_s driver interface starts interacting with net80211 module
1744          */
1745 #endif /* ! codereview */

1747         wpa_ctrl_close(ctrl_conn);
1724 out:
1725         if (command != NULL)
1726                 free(command);

1749         free(hexascii);
1728         return (status);
1729 }

1731 static dladm_status_t
1732 wpa_instance_delete(dladm_handle_t handle, datalink_id_t linkid)
1733 {
1734         char    instance_name[MAXLINKNAMELEN];

1736         /*
1737          * Get the instance name of the network/wpad service (the same as
1738          * the link name).
1739          */
1740         if (dladm_datalink_id2info(handle, linkid, NULL, NULL, NULL,
1741             instance_name, sizeof (instance_name)) != DLADM_STATUS_OK)
1742                 return (DLADM_STATUS_FAILED);

1751         return 0;
1744         return (delete_instance(instance_name));
1752 }
_____unchanged_portion_omitted_
```

****
12242 Tue Jun 12 19:54:57 2012
new/usr/src/lib/libdladm/common/libdlwlan.h
ess_list ioctl now provides all scan results properties for wpa/libdlwlan
first integration of wpa_s control interface client code
first integration of wpa_s wpa_ie parsing code
****

```
/*
 * 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
 */

#endif /* ! codereview */
/*
 * Copyright 2009 Sun Microsystems, Inc.  All rights reserved.
 * Use is subject to license terms.
 */

#ifndef _LIBDLWLAN_H
#define _LIBDLWLAN_H

/*
 * This file includes structures, macros and routines used by WLAN link
 * administration.
 */

#include <sys/types.h>
#include <sys/un.h>
#endif /* ! codereview */
#include <libdladm.h>

/*
 * General libdlwlan definitions and functions.
 *
 * These interfaces are ON consolidation-private.
 * For documentation, refer to PSARC/2006/623.
 */

#ifdef __cplusplus
extern "C" {
#endif

#define DLADM_WLAN_MAX_ESSID_LEN        32      /* per 802.11 spec */
#define DLADM_WLAN_MAX_ESSID_LEN    (32 + 1)    /* per 802.11 spec */
                                                /* max essid length is 32 */
                                                /* one more for '\0' */
#define DLADM_WLAN_BSSID_LEN            6       /* per 802.11 spec */
#define DLADM_WLAN_WPA_KEY_LEN          32      /* per 802.11i spec */
#define DLADM_WLAN_MAX_WPA_IE_LEN       40      /* per 802.11i spec */
```



```
#define DLADM_WLAN_CONNECT_TIMEOUT_DEFAULT      10
#define DLADM_WLAN_CONNECT_CREATEIBSS           0x00000001

#define IEEE80211_CAP_ESS                       0x0001
#define IEEE80211_CAP_IBSS                      0x0002
#define IEEE80211_CAP_PRIVACY                   0x0010
#define IEEE80211_CAP_RSN                       0x0800
#define IEEE80211_RATE                          0x7f
/* device driver capability, not net80211 ones*/
#define IEEE80211_C_WPA                         0x01800000

#define IEEE80211_CIPHER_WEP            0
#define IEEE80211_CIPHER_TKIP           1
#define IEEE80211_CIPHER_AES_OCB        2
#define IEEE80211_CIPHER_AES_CCM        3
#define IEEE80211_CIPHER_CKIP           4
#define IEEE80211_CIPHER_NONE           5       /* pseudo value */

#define IEEE80211_MLME_ASSOC            1       /* associate station */
#define IEEE80211_MLME_DISASSOC         2       /* disassociate station */
#define IEEE80211_MLME_DEAUTH           3       /* deauthenticate station */
#define IEEE80211_MLME_AUTHORIZE        4       /* authorize station */
#define IEEE80211_MLME_UNAUTHORIZE      5       /* unauthorize station */

/* Key Flags */
#define IEEE80211_KEY_XMIT              0x01    /* key used for xmit */
#define IEEE80211_KEY_RECV              0x02    /* key used for recv */
#define IEEE80211_KEY_GROUP             /* key used for WPA group operation */ \
                                        0x04
#define IEEE80211_KEY_SWCRYPT           0x10    /* host-based encrypt/decrypt */
#define IEEE80211_KEY_SWMIC             0x20    /* host-based enmic/demic */
#define IEEE80211_KEY_COMMON            /* common flags passed in by apps */ \
        (IEEE80211_KEY_XMIT | IEEE80211_KEY_RECV | IEEE80211_KEY_GROUP)

#define IEEE80211_KEY_DEFAULT           0x80    /* default xmit key */
#define DLADM_WLAN_CONNECT_NOSCAN               0x00000002

typedef struct dladm_wlan_essid {
        uint32_t we_length;
        uint8_t we_bytes[DLADM_WLAN_MAX_ESSID_LEN];
        char    we_bytes[DLADM_WLAN_MAX_ESSID_LEN];
} dladm_wlan_essid_t;
```

_____unchanged_portion_omitted_

```
typedef struct dladm_wlan_ess {
        dladm_wlan_bssid_t      we_bssid;
        dladm_wlan_essid_t      we_ssid;
        uint_t                  we_ssid_len;
        uint8_t                 we_wpa_ie[DLADM_WLAN_MAX_WPA_IE_LEN];
        uint_t                  we_wpa_ie_len;
        int                     we_freq;
} dladm_wlan_ess_t;
```

```
typedef enum {
        DLADM_WLAN_CIPHER_WEP = 0,
        DLADM_WLAN_CIPHER_TKIP,
        DLADM_WLAN_CIPHER_AES_OCB,
        DLADM_WLAN_CIPHER_AES_CCM,
        DLADM_WLAN_CIPHER_CKIP,
        DLADM_WLAN_CIPHER_NONE
} dladm_wlan_cipher_t;
```

_____unchanged_portion_omitted_

```
typedef enum {
        DLADM_WLAN_SECMODE_NONE = 1,
        DLADM_WLAN_SECMODE_WEP,
```

```
124         DLADM_WLAN_SECMODE_PSK,
125         DLADM_WLAN_SECMODE_EAP
 71         DLADM_WLAN_SECMODE_WPA
126 } dladm_wlan_secmode_t;
```

*unchanged_portion_omitted_*

```
145 /*
146  * auth_alg: list of allowed IEEE 802.11 authentication algorithms
147  * OPEN = Open System authentication (required for WPA/WPA2)
148  * SHARED = Shared Key authentication (requires extern WEP keys)
149  * an AP could have also no auth_alg. this is the only case where key_mgmt=NONE
150  * in wpa_s configuration should be used.
151  *
152  * when DLADM_WLAN_AUTH_NONE is set, DLADM_WLAN_SECMODE_NONE should be set too
153  */
154 #endif /* ! codereview */
155 typedef enum {
156         DLADM_WLAN_AUTH_NONE,
157 #endif /* ! codereview */
158         DLADM_WLAN_AUTH_OPEN = 1,
159         DLADM_WLAN_AUTH_SHARED
160 } dladm_wlan_auth_t;

162 typedef enum {
163         DLADM_WLAN_BSSTYPE_BSS = 0,
 91         DLADM_WLAN_BSSTYPE_BSS = 1,
164         DLADM_WLAN_BSSTYPE_IBSS,
165         DLADM_WLAN_BSSTYPE_AP
 93         DLADM_WLAN_BSSTYPE_ANY
166 } dladm_wlan_bsstype_t;
```

*unchanged_portion_omitted_*

```
173 typedef uint32_t dladm_wlan_speed_t;
174 typedef uint32_t dladm_wlan_channel_t;

176 /*
177  * wa_valid is a bitfield used for indicating the validity of each attribute.
178  * wa_valid may have 0 or more of the following bits set:
179  */
180 #endif /* ! codereview */
181 enum {
182         DLADM_WLAN_ATTR_ESSID = 0x00000001,
183         DLADM_WLAN_ATTR_BSSID = 0x00000002,
184         DLADM_WLAN_ATTR_SECMODE = 0x00000004,
185         DLADM_WLAN_ATTR_STRENGTH = 0x00000008,
186         DLADM_WLAN_ATTR_MODE = 0x00000010,
187         DLADM_WLAN_ATTR_SPEED = 0x00000020,
188         DLADM_WLAN_ATTR_AUTH = 0x00000040,
189         DLADM_WLAN_ATTR_BSSTYPE = 0x00000080,
190         DLADM_WLAN_ATTR_CHANNEL = 0x00000100
191 };

193 struct wpa_ie_data {
194         int proto;
195         int pairwise_cipher;
196         int group_cipher;
197         int key_mgmt;
198         int capabilities;
199         size_t num_pmkid;
200         const uint8_t *pmkid;
201         int mgmt_group_cipher;
202 };

204 #endif /* ! codereview */
205 typedef struct dladm_wlan_attr {
206         uint_t wa_valid;
```



```
207         dladm_wlan_essid_t wa_essid;
208         dladm_wlan_bssid_t wa_bssid;
209         dladm_wlan_secmode_t wa_secmode;
210         dladm_wlan_strength_t wa_strength;
211         dladm_wlan_mode_t wa_mode;
212         dladm_wlan_speed_t wa_speed;
213         dladm_wlan_auth_t wa_auth;
214         dladm_wlan_bsstype_t wa_bsstype;
215         dladm_wlan_channel_t wa_channel;
216         struct wpa_ie_data wa_ie;
217 #endif /* ! codereview */
218 } dladm_wlan_attr_t;

220 enum {
221         DLADM_WLAN_LINKATTR_STATUS = 0x00000001,
222         DLADM_WLAN_LINKATTR_WLAN = 0x00000002
223 };

225 #endif /* ! codereview */
226 typedef struct dladm_wlan_linkattr {
227         uint_t la_valid;
228         dladm_wlan_linkstatus_t la_status;
229         dladm_wlan_attr_t la_wlan_attr;
230 } dladm_wlan_linkattr_t;

232 #define DLADM_WLAN_WEPKEY64_LEN         5       /* per WEP spec */
233 #define DLADM_WLAN_WEPKEY128_LEN        13      /* per WEP spec */
234 #define DLADM_WLAN_MAX_KEY_LEN          64      /* per WEP/WPA spec */
235 #define DLADM_WLAN_MAX_WEPKEYS          4       /* MAX_NWEPKEYS */
236 #define DLADM_WLAN_MAX_KEYNAME_LEN      64

238 #endif /* ! codereview */
239 typedef struct dladm_wlan_key {
240         uint_t wk_idx;
241         uint_t wk_len;
242         uint8_t wk_val[DLADM_WLAN_MAX_KEY_LEN];
243         char wk_name[DLADM_WLAN_MAX_KEYNAME_LEN];
244         dladm_secobj_class_t wk_class;
104         uint_t          wk_class;
245 } dladm_wlan_key_t;

247 #define CTRL_IFACE_DIR          "/var/run/wpa_supplicant"
248 #define CTRL_IFACE_GLOBAL       "/var/run/wpa_supplicant-global"

250 /*
251  * struct wpa_ctrl - Internal structure for control interface library
252  *
253  * This structure is used by the wpa_supplicant/hostapd control interface
254  * library to store internal data. Programs using the library should not touch
255  * this data directly. They can only use the pointer to the data structure as
256  * an identifier for the control interface connection and use this as one of
257  * the arguments for most of the control interface library functions.
258  */
259 struct wpa_ctrl {
260         int s;
261         struct sockaddr_un local;
262         struct sockaddr_un dest;
263 };

265 #endif /* ! codereview */
266 extern dladm_status_t dladm_wlan_scan(dladm_handle_t, datalink_id_t, void *,
267     boolean_t(*)(void *, dladm_wlan_attr_t *));
107                             boolean_t (*)(void *, dladm_wlan_attr_t *));
268 extern dladm_status_t dladm_wlan_connect(dladm_handle_t, datalink_id_t,
269     dladm_wlan_attr_t *, int, void *, uint_t, uint_t, char *);
109                             dladm_wlan_attr_t *, int, void *, uint_t, uint_t);
```

```
270 extern dladm_status_t dladm_wlan_disconnect(dladm_handle_t, datalink_id_t);

272 /*GET*/
273 extern dladm_status_t dladm_wlan_get_linkattr(dladm_handle_t, datalink_id_t, dla

275 extern dladm_status_t dladm_wlan_get_essid(dladm_handle_t, datalink_id_t, void *
276 extern dladm_status_t dladm_wlan_get_bssid(dladm_handle_t, datalink_id_t, void *
277 extern dladm_status_t dladm_wlan_get_esslist(dladm_handle_t, datalink_id_t, void
278 extern dladm_status_t dladm_wlan_get_capability(dladm_handle_t, datalink_id_t, v

280 extern dladm_status_t   dladm_wlan_set_bsstype(dladm_handle_t, datalink_id_t,
281                             dladm_wlan_bsstype_t *);
282 extern dladm_status_t   dladm_wlan_set_authmode(dladm_handle_t, datalink_id_t,
283                             dladm_wlan_auth_t *);
284 extern dladm_status_t   dladm_wlan_set_encryption(dladm_handle_t, datalink_id_t,
285                             dladm_wlan_secmode_t *);
286 extern dladm_status_t   dladm_wlan_set_essid(dladm_handle_t, datalink_id_t,
287                             const uint8_t  *, size_t);
288 extern dladm_status_t   dladm_wlan_set_bssid(dladm_handle_t, datalink_id_t,
289                             dladm_wlan_bssid_t *);
290 extern dladm_status_t   dladm_wlan_set_createibss(dladm_handle_t, datalink_id_t,
291                             boolean_t *);
292 extern dladm_status_t   dladm_wlan_set_channel(dladm_handle_t, datalink_id_t,
293                             dladm_wlan_channel_t *);
111 extern dladm_status_t   dladm_wlan_get_linkattr(dladm_handle_t, datalink_id_t,
112                             dladm_wlan_linkattr_t *);
113 /* WPA support routines */
114 extern dladm_status_t   dladm_wlan_wpa_get_sr(dladm_handle_t, datalink_id_t,
115                             dladm_wlan_ess_t *, uint_t, uint_t *);
294 extern dladm_status_t dladm_wlan_wpa_set_ie(dladm_handle_t, datalink_id_t,
295     const uint8_t *, size_t);
117                             uint8_t *, uint_t);
296 extern dladm_status_t dladm_wlan_wpa_set_wpa(dladm_handle_t, datalink_id_t,
297     boolean_t);
298 extern dladm_status_t dladm_wlan_wpa_del_key(dladm_handle_t, datalink_id_t,
299     uint_t, const dladm_wlan_bssid_t *);
300 extern dladm_status_t dladm_wlan_wpa_set_key(dladm_handle_t, datalink_id_t,
301     dladm_wlan_cipher_t, const dladm_wlan_bssid_t *, boolean_t, uint64_t,
302     uint_t, uint8_t *, uint_t);
123                             dladm_wlan_cipher_t, const dladm_wlan_bssid_t *,
124                             boolean_t, uint64_t, uint_t, uint8_t *, uint_t);
303 extern dladm_status_t dladm_wlan_wpa_set_mlme(dladm_handle_t, datalink_id_t,
304     dladm_wlan_mlme_op_t, dladm_wlan_reason_t, const uint8_t *);
126                             dladm_wlan_mlme_op_t,
127                             dladm_wlan_reason_t, dladm_wlan_bssid_t *);

306 extern const char *dladm_wlan_essid2str(dladm_wlan_essid_t *, char *);
307 extern const char *dladm_wlan_bssid2str(dladm_wlan_bssid_t *, char *);
308 extern const char *dladm_wlan_secmode2str(dladm_wlan_secmode_t *, char *);
309 extern const char *dladm_wlan_strength2str(dladm_wlan_strength_t *, char *);
132 extern const char       *dladm_wlan_strength2str(dladm_wlan_strength_t *,
133                             char *);
310 extern const char *dladm_wlan_mode2str(dladm_wlan_mode_t *, char *);
311 extern const char *dladm_wlan_speed2str(dladm_wlan_speed_t *, char *);
312 extern const char *dladm_wlan_auth2str(dladm_wlan_auth_t *, char *);
313 extern const char *dladm_wlan_bsstype2str(dladm_wlan_bsstype_t *, char *);
314 extern const char *dladm_wlan_linkstatus2str(dladm_wlan_linkstatus_t *, char *);
138 extern const char       *dladm_wlan_linkstatus2str(dladm_wlan_linkstatus_t *,
139                             char *);

316 extern dladm_status_t dladm_wlan_str2essid(const char *, dladm_wlan_essid_t *);
317 extern dladm_status_t dladm_wlan_str2bssid(const char *, dladm_wlan_bssid_t *);
141 extern dladm_status_t   dladm_wlan_str2essid(const char *,
142                             dladm_wlan_essid_t *);
143 extern dladm_status_t   dladm_wlan_str2bssid(const char *,
144                             dladm_wlan_bssid_t *);
```



```
318 extern dladm_status_t dladm_wlan_str2secmode(const char *,
319     dladm_wlan_secmode_t *);
320 extern dladm_status_t dladm_wlan_str2strength(const char *,
321     dladm_wlan_strength_t *);
322 extern dladm_status_t dladm_wlan_str2mode(const char *, dladm_wlan_mode_t *);
323 extern dladm_status_t dladm_wlan_str2speed(const char *, dladm_wlan_speed_t *);
324 extern dladm_status_t dladm_wlan_str2auth(const char *, dladm_wlan_auth_t *);
149 extern dladm_status_t   dladm_wlan_str2mode(const char *,
150                             dladm_wlan_mode_t *);
151 extern dladm_status_t   dladm_wlan_str2speed(const char *,
152                             dladm_wlan_speed_t *);
153 extern dladm_status_t   dladm_wlan_str2auth(const char *,
154                             dladm_wlan_auth_t *);
325 extern dladm_status_t dladm_wlan_str2bsstype(const char *,
326     dladm_wlan_bsstype_t *);
327 extern dladm_status_t dladm_wlan_str2linkstatus(const char *,
328     dladm_wlan_linkstatus_t *);

330 /* wpa_ie parsing support routines */

332 /*
333  * wpa_parse_wpa_ie - Parse WPA/RSN IE
334  * @wpa_ie: Pointer to WPA or RSN IE
335  * @wpa_ie_len: Length of the WPA/RSN IE
336  * @data: Pointer to data area for parsing results
337  * Returns: 0 on success, -1 on failure
338  *
339  * Parse the contents of WPA or RSN IE and write the parsed data into data.
340  */
341 extern int wpa_parse_wpa_ie(const uint8_t *wpa_ie, size_t wpa_ie_len,
342     struct wpa_ie_data *data);

344 extern char * wpa_cipher_txt(int cipher);
345 extern char * wpa_key_mgmt_txt(int key_mgmt);

347 /* wpa_supplicant control interface client routines */

349 /*
350  * wpa_ctrl_open - Open a control interface to wpa_supplicant/hostapd
351  * @ctrl_path: Path for UNIX domain sockets;
352  * Returns: Pointer to abstract control interface data or %NULL on failure
353  */
354 extern struct wpa_ctrl * wpa_ctrl_open(const char *ctrl_path);

356 /*
357  * wpa_ctrl_close - Close a control interface to wpa_supplicant
358  * @ctrl: Control interface data from wpa_ctrl_open()
359  *
360  * This function is used to close a control interface.
361  */
362 extern void wpa_ctrl_close(struct wpa_ctrl *ctrl);

364 extern int wpa_request(struct wpa_ctrl *ctrl, int argc, char *argv[]);

366 #endif /* ! codereview */
367 #ifdef  __cplusplus
368 }
369 #endif

371 #endif  /* _LIBDLWLAN_H */
```

```
**********************************************************
    2939 Tue Jun 12 19:54:59 2012
new/usr/src/lib/libdladm/common/libdlwlan_impl.h
ess_list ioctl now provides all scan results properties for wpa/libdlwlan
first integration of wpa_s control interface client code
first integration of wpa_s wpa_ie parsing code
**********************************************************
______unchanged_portion_omitted_

 66 /*
 67  * not implemented:
 68  * powermode: Specifies the power management mode of the WiFi link.
 69  * Possible values are:
 70  * off (disable power management),
 71  * max (maximum power savings), and
 72  * fast (performance sensitive power management).
 73  * Default is off.
 74  *
 75  * radio: Specifies whether the radio is on or off; default is on.
 76  */

 78 #endif /* ! codereview */
 79 typedef enum {
 80         DLADM_WLAN_RADIO_ON = 1,
 81         DLADM_WLAN_RADIO_OFF
 82 } dladm_wlan_radio_t;

 84 typedef enum {
 85         DLADM_WLAN_PM_OFF = 1,
 86         DLADM_WLAN_PM_MAX,
 87         DLADM_WLAN_PM_FAST
 88 } dladm_wlan_powermode_t;

 90 extern  dladm_status_t i_dladm_wlan_legacy_ioctl(dladm_handle_t,
 91                             datalink_id_t, wldp_t *, uint_t, size_t, uint_t,
 92                             size_t);
 93 extern dladm_status_t   i_dladm_wlan_param(dladm_handle_t, datalink_id_t,
 94                             void *, mac_prop_id_t, size_t, boolean_t);
 95 extern boolean_t        i_dladm_wlan_convert_chan(wl_phy_conf_t *, uint32_t *);

 97 #ifdef  __cplusplus
 98 }
 99 #endif

101 #endif  /* _LIBDLWLAN_IMPL_H */
```

```
**********************************************************
  119693 Tue Jun 12 19:54:59 2012
new/usr/src/lib/libdladm/common/linkprop.c
ess_list ioctl now provides all scan results properties for wpa/libdlwlan
first integration of wpa_s control interface client code
first integration of wpa_s wpa_ie parsing code
**********************************************************
```

```
   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  */
  21 /*
  22  * Copyright (c) 2006, 2010, Oracle and/or its affiliates. All rights reserved.
  23  */

  25 #include <stdlib.h>
  26 #include <string.h>
  27 #include <strings.h>
  28 #include <errno.h>
  29 #include <ctype.h>
  30 #include <stddef.h>
  31 #include <sys/types.h>
  32 #include <sys/stat.h>
  33 #include <sys/dld.h>
  34 #include <sys/zone.h>
  35 #include <fcntl.h>
  36 #include <unistd.h>
  37 #include <libdevinfo.h>
  38 #include <zone.h>
  39 #include <libdllink.h>
  40 #include <libdladm_impl.h>
  41 #include <libdlwlan_impl.h>
  42 #include <libdlwlan.h>
  43 #include <libdlvlan.h>
  44 #include <libdlvnic.h>
  45 #include <libdlib.h>
  46 #include <libintl.h>
  47 #include <dlfcn.h>
  48 #include <link.h>
  49 #include <inet/wifi_ioctl.h>
  50 #include <libdladm.h>
  51 #include <libdlstat.h>
  52 #include <sys/param.h>
  53 #include <sys/debug.h>
  54 #include <sys/dld.h>
  55 #include <inttypes.h>
  56 #include <sys/ethernet.h>
  57 #include <inet/iptun.h>
  58 #include <net/wpa.h>
  58 #include <sys/sysmacros.h>
```



```
  59 #include <sys/vlan.h>
  60 #include <libdlbridge.h>
  61 #include <stp_in.h>
  62 #include <netinet/dhcp.h>
  63 #include <netinet/dhcp6.h>
  64 #include <net/if_types.h>
  65 #include <libinetutil.h>
  66 #include <pool.h>

  68 /*
  69  * The linkprop get() callback.
  70  * - pd:        pointer to the prop_desc_t
  71  * - propstrp:  a property string array to keep the returned property.
  72  *              Caller allocated.
  73  * - cntp:      number of returned properties.
  74  *              Caller also uses it to indicate how many it expects.
  75  */
  76 struct prop_desc;
  77 typedef struct prop_desc prop_desc_t;

  79 typedef dladm_status_t  pd_getf_t(dladm_handle_t, prop_desc_t *pdp,
  80                         datalink_id_t, char **propstp, uint_t *cntp,
  81                         datalink_media_t, uint_t, uint_t *);

  83 /*
  84  * The linkprop set() callback.
  85  * - propval:   a val_desc_t array which keeps the property values to be set.
  86  * - cnt:       number of properties to be set.
  87  * - flags:     additional flags passed down the system call.
  88  *
  89  * pd_set takes val_desc_t given by pd_check(), translates it into
  90  * a format suitable for kernel consumption. This may require allocation
  91  * of ioctl buffers etc. pd_set() may call another common routine (used
  92  * by all other pd_sets) which invokes the ioctl.
  93  */
  94 typedef dladm_status_t  pd_setf_t(dladm_handle_t, prop_desc_t *, datalink_id_t,
  95                             val_desc_t *propval, uint_t cnt, uint_t flags,
  96                             datalink_media_t);

  98 /*
  99  * The linkprop check() callback.
 100  * - propstrp:  property string array which keeps the property to be checked.
 101  * - cnt:       number of properties.
 102  * - propval:   return value; the property values of the given property strings.
 103  *
 104  * pd_check checks that the input values are valid. It does so by
 105  * iteraring through the pd_modval list for the property. If
 106  * the modifiable values cannot be expressed as a list, a pd_check
 107  * specific to this property can be used. If the input values are
 108  * verified to be valid, pd_check allocates a val_desc_t and fills it
 109  * with either a val_desc_t found on the pd_modval list or something
 110  * generated on the fly.
 111  */
 112 typedef dladm_status_t  pd_checkf_t(dladm_handle_t, prop_desc_t *pdp,
 113                             datalink_id_t, char **propstrp, uint_t *cnt,
 114                             uint_t flags, val_desc_t **propval,
 115                             datalink_media_t);

 117 typedef struct link_attr_s {
 118         mac_prop_id_t   pp_id;
 119         size_t          pp_valsize;
 120         char            *pp_name;
 121 } link_attr_t;
_____unchanged_portion_omitted_

 229 #define MAC_PROP_BUFSIZE(v)     sizeof (dld_ioc_macprop_t) + (v) - 1
```

```
231 /*
232  * Supported link properties enumerated in the prop_table[] array are
233  * computed using the callback functions in that array. To compute the
234  * property value, multiple distinct system calls may be needed (e.g.,
235  * for wifi speed, we need to issue system calls to get desired/supported
236  * rates). The link_attr[] table enumerates the interfaces to the kernel,
237  * and the type/size of the data passed in the user-kernel interface.
238  */
239 static link_attr_t link_attr[] = {
240         { MAC_PROP_DUPLEX,      sizeof (link_duplex_t), "duplex"},

242         { MAC_PROP_SPEED,       sizeof (uint64_t),      "speed"},

244         { MAC_PROP_STATUS,      sizeof (link_state_t),  "state"},

246         { MAC_PROP_AUTONEG,     sizeof (uint8_t),       "adv_autoneg_cap"},

248         { MAC_PROP_MTU,         sizeof (uint32_t),      "mtu"},

250         { MAC_PROP_FLOWCTRL,    sizeof (link_flowctrl_t), "flowctrl"},

252         { MAC_PROP_ZONE,        sizeof (dld_ioc_zid_t), "zone"},

254         { MAC_PROP_AUTOPUSH,    sizeof (struct dlautopush), "autopush"},

256         { MAC_PROP_ADV_10GFDX_CAP, sizeof (uint8_t),    "adv_10gfdx_cap"},

258         { MAC_PROP_EN_10GFDX_CAP, sizeof (uint8_t),     "en_10gfdx_cap"},

260         { MAC_PROP_ADV_1000FDX_CAP, sizeof (uint8_t),   "adv_1000fdx_cap"},

262         { MAC_PROP_EN_1000FDX_CAP, sizeof (uint8_t),    "en_1000fdx_cap"},

264         { MAC_PROP_ADV_1000HDX_CAP, sizeof (uint8_t),   "adv_1000hdx_cap"},

266         { MAC_PROP_EN_1000HDX_CAP, sizeof (uint8_t),    "en_1000hdx_cap"},

268         { MAC_PROP_ADV_100FDX_CAP, sizeof (uint8_t),    "adv_100fdx_cap"},

270         { MAC_PROP_EN_100FDX_CAP, sizeof (uint8_t),     "en_100fdx_cap"},

272         { MAC_PROP_ADV_100HDX_CAP, sizeof (uint8_t),    "adv_100hdx_cap"},

274         { MAC_PROP_EN_100HDX_CAP, sizeof (uint8_t),     "en_100hdx_cap"},

276         { MAC_PROP_ADV_10FDX_CAP, sizeof (uint8_t),     "adv_10fdx_cap"},

278         { MAC_PROP_EN_10FDX_CAP, sizeof (uint8_t),      "en_10fdx_cap"},

280         { MAC_PROP_ADV_10HDX_CAP, sizeof (uint8_t),     "adv_10hdx_cap"},

282         { MAC_PROP_EN_10HDX_CAP, sizeof (uint8_t),      "en_10hdx_cap"},

284         { MAC_PROP_WL_ESSID,    sizeof (wl_linkstatus_t), "essid"},

286         { MAC_PROP_WL_BSSID,    sizeof (wl_bssid_t),    "bssid"},

288         { MAC_PROP_WL_BSSTYPE,  sizeof (wl_bss_type_t), "bsstype"},

290         { MAC_PROP_WL_LINKSTATUS, sizeof (wl_linkstatus_t), "wl_linkstatus"},

292         /* wl_rates_t has variable length */
293         { MAC_PROP_WL_DESIRED_RATES, sizeof (wl_rates_t), "desired_rates"},

295         /* wl_rates_t has variable length */
```



```
296         { MAC_PROP_WL_SUPPORTED_RATES, sizeof (wl_rates_t), "supported_rates"},

298         { MAC_PROP_WL_AUTH_MODE, sizeof (wl_authmode_t), "authmode"},

300         { MAC_PROP_WL_ENCRYPTION, sizeof (wl_encryption_t), "encryption"},

302         { MAC_PROP_WL_RSSI,     sizeof (wl_rssi_t),     "signal"},

304         { MAC_PROP_WL_PHY_CONFIG, sizeof (wl_phy_conf_t), "phy_conf"},

306         { MAC_PROP_WL_CAPABILITY, sizeof (wl_capability_t), "capability"},

308         { MAC_PROP_WL_WPA,      sizeof (wl_wpa_t),      "wpa"},

311         /*  wl_wpa_ess_t has variable length */
312         { MAC_PROP_WL_SCANRESULTS, sizeof (wl_wpa_ess_t), "scan_results"},

310         { MAC_PROP_WL_POWER_MODE, sizeof (wl_ps_mode_t), "powermode"},

312         { MAC_PROP_WL_RADIO,    sizeof (dladm_wlan_radio_t), "wl_radio"},

314         { MAC_PROP_WL_ESS_LIST, sizeof (wl_ess_list_t), "wl_ess_list"},

316         { MAC_PROP_WL_KEY_TAB,  sizeof (wl_wep_key_tab_t), "wl_wep_key"},

318         { MAC_PROP_WL_CREATE_IBSS, sizeof (wl_create_ibss_t), "createibss"},

320         /* wl_wpa_ie_t has variable length */
321         { MAC_PROP_WL_SETOPTIE, sizeof (wl_wpa_ie_t),   "set_ie"},

323         { MAC_PROP_WL_DELKEY,   sizeof (wl_del_key_t),  "wpa_del_key"},

325         { MAC_PROP_WL_KEY,      sizeof (wl_key_t),      "wl_key"},

327         { MAC_PROP_WL_MLME,     sizeof (wl_mlme_t),     "mlme"},

329         { MAC_PROP_TAGMODE,     sizeof (link_tagmode_t),        "tagmode"},

331         { MAC_PROP_IPTUN_HOPLIMIT, sizeof (uint32_t),   "hoplimit"},

333         { MAC_PROP_IPTUN_ENCAPLIMIT, sizeof (uint32_t), "encaplimit"},

335         { MAC_PROP_PVID,        sizeof (uint16_t),      "default_tag"},

337         { MAC_PROP_LLIMIT,      sizeof (uint32_t),      "learn_limit"},

339         { MAC_PROP_LDECAY,      sizeof (uint32_t),      "learn_decay"},

341         { MAC_PROP_RESOURCE,    sizeof (mac_resource_props_t),  "resource"},

343         { MAC_PROP_RESOURCE_EFF, sizeof (mac_resource_props_t),
344             "resource-effective"},

346         { MAC_PROP_RXRINGSRANGE, sizeof (mac_propval_range_t),  "rxrings"},

348         { MAC_PROP_TXRINGSRANGE, sizeof (mac_propval_range_t),  "txrings"},

350         { MAC_PROP_MAX_TX_RINGS_AVAIL,  sizeof (uint_t),
351             "txrings-available"},

353         { MAC_PROP_MAX_RX_RINGS_AVAIL,  sizeof (uint_t),
354             "rxrings-available"},

356         { MAC_PROP_MAX_RXHWCLNT_AVAIL,  sizeof (uint_t), "rxhwclnt-available"},

358         { MAC_PROP_MAX_TXHWCLNT_AVAIL,  sizeof (uint_t), "txhwclnt-available"},
```

```
 360         { MAC_PROP_IB_LINKMODE, sizeof (uint32_t),      "linkmode"},

 362         { MAC_PROP_PRIVATE,     0,                      "driver-private"}
 363 };
```

_____***unchanged_portion_omitted_***

```
**********************************************************
    6340 Tue Jun 12 19:55:03 2012
new/usr/src/lib/libdladm/common/mapfile-vers
cleaned lint warnings
**********************************************************
#
# 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) 2006, 2010, Oracle and/or its affiliates. All rights reserved.
#

#
# MAPFILE HEADER START
#
# WARNING:  STOP NOW.  DO NOT MODIFY THIS FILE.
# Object versioning must comply with the rules detailed in
#
#       usr/src/lib/README.mapfiles
#
# You should not be making modifications here until you've read the most current
# copy of that file. If you need help, contact a gatekeeper for guidance.
#
# MAPFILE HEADER END
#

$mapfile_version 2

SYMBOL_VERSION SUNWprivate_1.1 {
    global:
        dladm_open;
        dladm_close;
        dladm_dld_fd;
        dladm_door_fd;
        dladm_info;
        dladm_walk;
        dladm_status2str;
        dladm_linkstate2str;
        dladm_linkduplex2str;
        dladm_set_rootdir;
        dladm_valid_linkname;
        dladm_mac_walk;
        dladm_init_linkprop;
        dladm_get_linkprop;
        dladm_get_linkprop_values;
        dladm_set_linkprop;
        dladm_walk_linkprop;
        dladm_attr_is_linkprop;
        dladm_linkprop_is_set;
```



```
        dladm_valid_secobj_name;
        dladm_init_secobj;
        dladm_get_secobj;
        dladm_set_secobj;
        dladm_unset_secobj;
        dladm_walk_secobj;
        dladm_str2interval;
        dladm_bw2str;
        dladm_str2bw;
        dladm_secobjclass2str;
        dladm_str2secobjclass;
        dladm_aggr_up;
        dladm_aggr_add;
        dladm_aggr_create;
        dladm_aggr_delete;
        dladm_aggr_modify;
        dladm_aggr_remove;
        dladm_aggr_lacpmode2str;
        dladm_aggr_lacptimer2str;
        dladm_aggr_macaddr2str;
        dladm_aggr_policy2str;
        dladm_aggr_portstate2str;
        dladm_aggr_str2lacpmode;
        dladm_aggr_str2lacptimer;
        dladm_aggr_str2macaddr;
        dladm_aggr_str2policy;
        dladm_aggr_info;
        dladm_key2linkid;
        dladm_wlan_scan;
        dladm_wlan_connect;
        dladm_wlan_disconnect;
        dladm_wlan_get_bssid;
        dladm_wlan_get_essid;
        dladm_wlan_get_esslist;
#endif /* ! codereview */
        dladm_wlan_get_linkattr;
        dladm_wlan_set_authmode;
        dladm_wlan_set_bsstype;
        dladm_wlan_set_channel;
        dladm_wlan_set_encryption;
        dladm_wlan_set_essid;
#endif /* ! codereview */
        dladm_wlan_essid2str;
        dladm_wlan_bssid2str;
        dladm_wlan_secmode2str;
        dladm_wlan_strength2str;
        dladm_wlan_mode2str;
        dladm_wlan_speed2str;
        dladm_wlan_auth2str;
        dladm_wlan_bsstype2str;
        dladm_wlan_linkstatus2str;
        dladm_wlan_str2essid;
        dladm_wlan_str2bssid;
        dladm_wlan_str2secmode;
        dladm_wlan_str2strength;
        dladm_wlan_str2mode;
        dladm_wlan_str2speed;
        dladm_wlan_str2auth;
        dladm_wlan_str2bsstype;
        dladm_wlan_str2linkstatus;
        dladm_iptun_create;
        dladm_iptun_delete;
        dladm_iptun_modify;
        dladm_iptun_getparams;
        dladm_iptun_up;
        dladm_iptun_down;
```

```
128         dladm_iptun_set6to4relay;
129         dladm_iptun_get6to4relay;
130         dladm_vlan_create;
131         dladm_vlan_delete;
132         dladm_vlan_up;
133         dladm_vlan_info;
134         dladm_class2str;
135         dladm_media2str;
136         dladm_str2media;
137         dladm_rename_link;
138         dladm_phys_info;
139         dladm_phys_delete;
140         dladm_dev2linkid;
141         dladm_linkid2legacyname;
142         dladm_create_datalink_id;
143         dladm_destroy_datalink_id;
144         dladm_remap_datalink_id;
145         dladm_up_datalink_id;
146         dladm_name2info;
147         dladm_datalink_id2info;
148         dladm_walk_datalink_id;
149         dladm_create_conf;
150         dladm_open_conf;
151         dladm_getsnap_conf;
152         dladm_write_conf;
153         dladm_remove_conf;
154         dladm_destroy_conf;
155         dladm_get_conf_field;
156         dladm_set_conf_field;
157         dladm_unset_conf_field;
 93         dladm_wlan_wpa_get_sr;
158         dladm_wlan_wpa_set_ie;
159         dladm_wlan_wpa_set_wpa;
160         dladm_wlan_wpa_del_key;
161         dladm_wlan_wpa_set_key;
162         dladm_wlan_wpa_set_mlme;
163         dladm_vnic_create;
164         dladm_vnic_delete;
165         dladm_vnic_info;
166         dladm_vnic_str2macaddrtype;
167         dladm_vnic_up;
168         dladm_walk_macaddr;
169         dladm_walk_hwgrp;
170         dladm_pri2str;
171         dladm_str2pri;
172         dladm_protect2str;
173         dladm_str2protect;
174         dladm_ipv4addr2str;
175         dladm_str2ipv4addr;
176         dladm_ipv6addr2str;
177         dladm_str2ipv6addr;
178         dladm_start_usagelog;
179         dladm_stop_usagelog;
180         dladm_walk_usage_res;
181         dladm_walk_usage_time;
182         dladm_usage_summary;
183         dladm_usage_dates;
184         dladm_zone_boot;
185         dladm_zone_halt;

187         dladm_flow_add;
188         dladm_flow_remove;
189         dladm_flow_parse_db;
190         dladm_walk_flow;
191         dladm_flow_init;
192         dladm_flow_info;
```



```
193         dladm_prefixlen2mask;
194         dladm_mask2prefixlen;
195         dladm_str2proto;
196         dladm_proto2str;

198         dladm_free_attrs;
199         dladm_parse_flow_attrs;

201         dladm_flow_attr_ip2str;
202         dladm_flow_attr_proto2str;
203         dladm_flow_attr_port2str;
204         dladm_flow_attr_dsfield2str;

206         dladm_free_props;
207         dladm_parse_link_props;
208         dladm_get_linkprop;
209         dladm_set_linkprop;
210         dladm_walk_linkprop;
211         dladm_parse_flow_props;
212         dladm_get_flowprop;
213         dladm_set_flowprop;
214         dladm_walk_flowprop;

216         dladm_parselink;

218         dladm_continuous;
219         dladm_kstat_lookup;
220         dladm_get_stats;
221         dladm_kstat_value;
222         dladm_get_single_mac_stat;
223         dladm_stats_total;
224         dladm_stats_diff;
225         dladm_ether_info;
226         dladm_ether_autoneg2str;
227         dladm_ether_pause2str;
228         dladm_ether_spdx2str;
229         dladm_ether_info_done;

231         dladm_simnet_create;
232         dladm_simnet_modify;
233         dladm_simnet_delete;
234         dladm_simnet_info;
235         dladm_simnet_up;
236         dladm_bridge_str2prot;
237         dladm_bridge_prot2str;
238         dladm_bridge_get_properties;
239         dladm_bridge_run_properties;
240         dladm_bridge_configure;
241         dladm_bridge_enable;
242         dladm_bridge_delete;
243         dladm_bridge_state;
244         dladm_bridge_get_portlist;
245         dladm_bridge_free_portlist;
246         dladm_bridge_setlink;
247         dladm_bridge_getlink;
248         dladm_bridge_link_state;
249         dladm_valid_bridgename;
250         dladm_observe_to_bridge;
251         dladm_bridge_get_fwdtable;
252         dladm_bridge_free_fwdtable;
253         dladm_bridge_get_trillnick;
254         dladm_bridge_free_trillnick;
255         dladm_bridge_get_nick;
256         dladm_bridge_set_nick;
257         dladm_bridge_get_privprop;
```

```
 259         dladm_link_stat_query;
 260         dladm_link_stat_diffchain;
 261         dladm_link_stat_free;
 262         dladm_link_stat_query_all;
 263         dladm_link_stat_query_all_free;

 265         dladm_flow_stat_query;
 266         dladm_flow_stat_diff;
 267         dladm_flow_stat_free;
 268         dladm_flow_stat_query_all;
 269         dladm_flow_stat_query_all_free;

 271         dladm_part_create;
 272         dladm_part_delete;
 273         dladm_part_info;
 274         dladm_part_up;
 275         dladm_ib_info;
 276         dladm_free_ib_info;

 278         dladm_range2strs;
 279         dladm_strs2range;
 280         dladm_range2list;
 281         dladm_list2range;
 282     local:
 283         *;
 284 };
```

*unchanged_portion_omitted_*

****
17009 Tue Jun 12 19:55:04 2012
new/usr/src/lib/libdladm/common/secobj.c
secobjs types now are "wep, psk, eap, pin"
dladm_wlan_secmode_t and dladm_secobj_class_t are not related anymore
****
_____***unchanged_portion_omitted_***

```
 53 static secobj_class_info_t secobj_class_table[] = {
 54         {"wep",          DLD_SECOBJ_CLASS_WEP},
 55         {"psk",          DLD_SECOBJ_CLASS_PSK},
 56         {"eap-tls",      DLD_SECOBJ_CLASS_TLS},
 57         {"eap-ttls",     DLD_SECOBJ_CLASS_TTLS},
 58         {"peap",         DLD_SECOBJ_CLASS_PEAP}
 55         {"wpa", DLD_SECOBJ_CLASS_WPA}
 59 };
```

_____***unchanged_portion_omitted_***

```
**********************************************************
   17925 Tue Jun 12 19:55:06 2012
new/usr/src/lib/libdladm/common/wpa_ie.c
secobjs types now are "wep, psk, eap, pin"
dladm_wlan_secmode_t and dladm_secobj_class_t are not related anymore
**********************************************************
```

```
/*
 * Copyright (c) 2002-2012, Jouni Malinen <j@w1.fi>
 *
 * This program is free software; you can redistribute it and/or modify
 * it under the terms of the GNU General Public License version 2 as
 * published by the Free Software Foundation.
 *
 * Alternatively, this software may be distributed under the terms of BSD
 * license.
 */

#include <unistd.h>
#include <stdlib.h>
#include <stdio.h>
#include <strings.h>
#include <errno.h>
#include <ctype.h>
#include <fcntl.h>
#include <libdlwlan.h>

/* IEEE 802.11i */
#define PMKID_LEN 16

#define WPA_SELECTOR_LEN 4
#define WPA_VERSION 1
#define RSN_SELECTOR_LEN 4
#define RSN_VERSION 1

#define BIT(x) (1 << (x))

/* key_mgmt: */
#define WPA_KEY_MGMT_IEEE8021X BIT(0)
#define WPA_KEY_MGMT_PSK BIT(1)
#define WPA_KEY_MGMT_NONE BIT(2)
#define WPA_KEY_MGMT_IEEE8021X_NO_WPA BIT(3)
#define WPA_KEY_MGMT_WPA_NONE BIT(4)

/* proto: */
#define WPA_PROTO_WPA BIT(0)
#define WPA_PROTO_RSN BIT(1)

/* pairwise: and group: cyphers */
#define WPA_CIPHER_NONE BIT(0)
#define WPA_CIPHER_WEP40 BIT(1)
#define WPA_CIPHER_WEP104 BIT(2)
#define WPA_CIPHER_TKIP BIT(3)
#define WPA_CIPHER_CCMP BIT(4)

/* not supported by net80211*/
#define WPA_KEY_MGMT_FT_IEEE8021X BIT(5)
#define WPA_KEY_MGMT_FT_PSK BIT(6)
#define WPA_KEY_MGMT_IEEE8021X_SHA256 BIT(7)
#define WPA_KEY_MGMT_PSK_SHA256 BIT(8)
#define WPA_KEY_MGMT_WPS BIT(9)

/* Information Element IDs */
#define WLAN_EID_SSID 0
#define WLAN_EID_RSN 48
#define WLAN_EID_VENDOR_SPECIFIC 221
```



```
#define WPA_GET_LE16(a) ((uint16_t) (((a)[1] << 8) | (a)[0]))
#define WPA_GET_BE16(a) ((uint16_t) (((a)[0] << 8) | (a)[1]))
#define WPA_PUT_BE16(a, val)                    \
        do {                                    \
                (a)[0] = ((uint16_t) (val)) >> 8;    \
                (a)[1] = ((uint16_t) (val)) & 0xff;  \
        } while (0)


#define RSN_SELECTOR(a, b, c, d) \
        ((((uint32_t) (a)) << 24) | (((uint32_t) (b)) << 16) | (((uint32_t) (c)) \
         (uint32_t) (d))

#define WPA_AUTH_KEY_MGMT_NONE RSN_SELECTOR(0x00, 0x50, 0xf2, 0)
#define WPA_AUTH_KEY_MGMT_UNSPEC_802_1X RSN_SELECTOR(0x00, 0x50, 0xf2, 1)
#define WPA_AUTH_KEY_MGMT_PSK_OVER_802_1X RSN_SELECTOR(0x00, 0x50, 0xf2, 2)
#define WPA_CIPHER_SUITE_NONE RSN_SELECTOR(0x00, 0x50, 0xf2, 0)
#define WPA_CIPHER_SUITE_WEP40 RSN_SELECTOR(0x00, 0x50, 0xf2, 1)
#define WPA_CIPHER_SUITE_TKIP RSN_SELECTOR(0x00, 0x50, 0xf2, 2)
#define WPA_CIPHER_SUITE_WRAP RSN_SELECTOR(0x00, 0x50, 0xf2, 3)
#define WPA_CIPHER_SUITE_CCMP RSN_SELECTOR(0x00, 0x50, 0xf2, 4)
#define WPA_CIPHER_SUITE_WEP104 RSN_SELECTOR(0x00, 0x50, 0xf2, 5)


#define RSN_AUTH_KEY_MGMT_UNSPEC_802_1X RSN_SELECTOR(0x00, 0x0f, 0xac, 1)
#define RSN_AUTH_KEY_MGMT_PSK_OVER_802_1X RSN_SELECTOR(0x00, 0x0f, 0xac, 2)

#define RSN_AUTH_KEY_MGMT_802_1X_SHA256 RSN_SELECTOR(0x00, 0x0f, 0xac, 5)
#define RSN_AUTH_KEY_MGMT_PSK_SHA256 RSN_SELECTOR(0x00, 0x0f, 0xac, 6)
#define RSN_AUTH_KEY_MGMT_TPK_HANDSHAKE RSN_SELECTOR(0x00, 0x0f, 0xac, 7)

#define RSN_CIPHER_SUITE_NONE RSN_SELECTOR(0x00, 0x0f, 0xac, 0)
#define RSN_CIPHER_SUITE_WEP40 RSN_SELECTOR(0x00, 0x0f, 0xac, 1)
#define RSN_CIPHER_SUITE_TKIP RSN_SELECTOR(0x00, 0x0f, 0xac, 2)
#define RSN_CIPHER_SUITE_WRAP RSN_SELECTOR(0x00, 0x0f, 0xac, 3
#define RSN_CIPHER_SUITE_CCMP RSN_SELECTOR(0x00, 0x0f, 0xac, 4)
#define RSN_CIPHER_SUITE_WEP104 RSN_SELECTOR(0x00, 0x0f, 0xac, 5)
#define RSN_CIPHER_SUITE_AES_128_CMAC RSN_SELECTOR(0x00, 0x0f, 0xac, 6)

#define WPA_OUI_TYPE RSN_SELECTOR(0x00, 0x50, 0xf2, 1)

#define RSN_SELECTOR_PUT(a, val) WPA_PUT_BE32((uint8_t *) (a), (val))
#define RSN_SELECTOR_GET(a) WPA_GET_BE32((const uint8_t *) (a))

#define RSN_NUM_REPLAY_COUNTERS_1 0
#define RSN_NUM_REPLAY_COUNTERS_2 1
#define RSN_NUM_REPLAY_COUNTERS_4 2
#define RSN_NUM_REPLAY_COUNTERS_16 3

/* IEEE 802.11, 7.3.2.25.3 RSN Capabilities */
#define WPA_CAPABILITY_PREAUTH BIT(0)
#define WPA_CAPABILITY_NO_PAIRWISE BIT(1)

#define WPA_GET_BE32(a) ((((uint32_t) (a)[0]) << 24) | (((uint32_t) (a)[1]) << 1
                         (((uint32_t) (a)[2]) << 8) | ((uint32_t) (a)[3]))

/* B14-B15: Reserved */

/* WPA IE version 1
 * 00-50-f2:1 (OUI:OUI type)
 * 0x01 0x00 (version; little endian)
 * (all following fields are optional:)
 * Group Suite Selector (4 octets) (default: TKIP)
 * Pairwise Suite Count (2 octets, little endian) (default: 1)
 * Pairwise Suite List (4 * n octets) (default: TKIP)
 * Authenticated Key Management Suite Count (2 octets, little endian)
```

```
 *    (default: 1)
 * Authenticated Key Management Suite List (4 * n octets)
 *    (default: unspec 802.1X)
 * WPA Capabilities (2 octets, little endian) (default: 0)
 */

#pragma pack(1)
struct wpa_ie_hdr {
        uint8_t elem_id;
        uint8_t len;
        uint8_t oui[4]; /* 24-bit OUI followed by 8-bit OUI type */
        uint8_t version[2]; /* little endian */
};
#pragma pack()

/* 1/4: PMKID
 * 2/4: RSN IE
 * 3/4: one or two RSN IEs + GTK IE (encrypted)
 * 4/4: empty
 * 1/2: GTK IE (encrypted)
 * 2/2: empty
 */

/* RSN IE version 1
 * 0x01 0x00 (version; little endian)
 * (all following fields are optional:)
 * Group Suite Selector (4 octets) (default: CCMP)
 * Pairwise Suite Count (2 octets, little endian) (default: 1)
 * Pairwise Suite List (4 * n octets) (default: CCMP)
 * Authenticated Key Management Suite Count (2 octets, little endian)
 *    (default: 1)
 * Authenticated Key Management Suite List (4 * n octets)
 *    (default: unspec 802.1X)
 * RSN Capabilities (2 octets, little endian) (default: 0)
 * PMKID Count (2 octets) (default: 0)
 * PMKID List (16 * n octets)
 * Management Group Cipher Suite (4 octets) (default: AES-128-CMAC)
 */

#pragma pack(1)
struct rsn_ie_hdr {
        uint8_t elem_id; /* WLAN_EID_RSN */
        uint8_t len;
        uint8_t version[2]; /* little endian */
};
#pragma pack()

/* local functions prototypes */

static int wpa_parse_wpa_ie_wpa(const uint8_t *wpa_ie, size_t wpa_ie_len,
                        struct wpa_ie_data *data);
static int wpa_parse_wpa_ie_rsn(const uint8_t *rsn_ie, size_t rsn_ie_len,
                        struct wpa_ie_data *data);
/**
 * wpa_cipher_txt - Convert cipher suite to a text string
 * @cipher: Cipher suite (WPA_CIPHER_* enum)
 * Returns: Pointer to a text string of the cipher suite name
 */
char * wpa_cipher_txt(int cipher)
{
        switch (cipher) {
        case WPA_CIPHER_NONE:
                return "NONE";
        case WPA_CIPHER_WEP40:
                return "WEP-40";
        case WPA_CIPHER_WEP104:
```



```
                return "WEP-104";
        case WPA_CIPHER_TKIP:
                return "TKIP";
        case WPA_CIPHER_CCMP:
                return "CCMP";
        default:
                return "UNKNOWN";
        }
}

/**
 * wpa_key_mgmt_txt - Convert key management suite to a text string
 * @key_mgmt: Key management suite (WPA_KEY_MGMT_* enum)
 * @proto: WPA/WPA2 version (WPA_PROTO_*)
 * Returns: Pointer to a text string of the key management suite name
 */
char * wpa_key_mgmt_txt(int key_mgmt)
{
        switch (key_mgmt) {
        case WPA_KEY_MGMT_IEEE8021X:
                return "WPA-EAP";
        case WPA_KEY_MGMT_PSK:
                return "WPA-PSK";
        case WPA_KEY_MGMT_NONE:
                return "NONE";
        case WPA_KEY_MGMT_IEEE8021X_NO_WPA:
                return "IEEE8021X";
        default:
                return "UNKNOWN";
        }
}

static int wpa_cypher_to_bitfield(const uint8_t *s)
{
        if (RSN_SELECTOR_GET(s) == WPA_CIPHER_SUITE_NONE)
                return WPA_CIPHER_NONE;
        if (RSN_SELECTOR_GET(s) == WPA_CIPHER_SUITE_WEP40)
                return WPA_CIPHER_WEP40;
        if (RSN_SELECTOR_GET(s) == WPA_CIPHER_SUITE_TKIP)
                return WPA_CIPHER_TKIP;
        if (RSN_SELECTOR_GET(s) == WPA_CIPHER_SUITE_CCMP)
                return WPA_CIPHER_CCMP;
        if (RSN_SELECTOR_GET(s) == WPA_CIPHER_SUITE_WEP104)
                return WPA_CIPHER_WEP104;
        return 0;
}

static int wpa_key_mgmt_to_bitfield(const uint8_t *s)
{
        if (RSN_SELECTOR_GET(s) == WPA_AUTH_KEY_MGMT_UNSPEC_802_1X)
                return WPA_KEY_MGMT_IEEE8021X;
        if (RSN_SELECTOR_GET(s) == WPA_AUTH_KEY_MGMT_PSK_OVER_802_1X)
                return WPA_KEY_MGMT_PSK;
        if (RSN_SELECTOR_GET(s) == WPA_AUTH_KEY_MGMT_NONE)
                return WPA_KEY_MGMT_WPA_NONE;
        return 0;
}

static int wpa_parse_wpa_ie_wpa(const uint8_t *wpa_ie, size_t wpa_ie_len,
                        struct wpa_ie_data *data)
{
        const struct wpa_ie_hdr *hdr;
        const uint8_t *pos;
        int left;
        int i, count;
```

```
259         memset(data, 0, sizeof(*data));
260         data->proto = WPA_PROTO_WPA;
261         data->pairwise_cipher = WPA_CIPHER_TKIP;
262         data->group_cipher = WPA_CIPHER_TKIP;
263         data->key_mgmt = WPA_KEY_MGMT_IEEE8021X;
264         data->capabilities = 0;
265         data->pmkid = NULL;
266         data->num_pmkid = 0;
267         data->mgmt_group_cipher = 0;

269         if (wpa_ie_len == 0) {
270                 /* No WPA IE - fail silently */
271                 return -1;
272         }

274         if (wpa_ie_len < sizeof(struct wpa_ie_hdr)) {
275                 printf("wpa_parse_wpa_ie_wpa: ie len too short %lu",
276                     (unsigned long) wpa_ie_len);
277                 return -1;
278         }

280         hdr = (const struct wpa_ie_hdr *) wpa_ie;

282         if (hdr->elem_id != WLAN_EID_VENDOR_SPECIFIC ||
283             hdr->len != wpa_ie_len - 2 ||
284             RSN_SELECTOR_GET(hdr->oui) != WPA_OUI_TYPE ||
285             WPA_GET_LE16(hdr->version) != WPA_VERSION) {
286                 printf("wpa_parse_wpa_ie_wpa: malformed ie or unknown version");
287                 return -2;
288         }

290         pos = (const uint8_t *) (hdr + 1);
291         left = wpa_ie_len - sizeof(*hdr);

293         if (left >= WPA_SELECTOR_LEN) {
294                 data->group_cipher = wpa_cypher_to_bitfield(pos);
295                 pos += WPA_SELECTOR_LEN;
296                 left -= WPA_SELECTOR_LEN;
297         } else if (left > 0) {
298                 printf("wpa_parse_wpa_ie_wpa: ie length mismatch, %u too much",
299                     left);
300                 return -3;
301         }

303         if (left >= 2) {
304                 data->pairwise_cipher = 0;
305                 count = WPA_GET_LE16(pos);
306                 pos += 2;
307                 left -= 2;
308                 if (count == 0 || left < count * WPA_SELECTOR_LEN) {
309                         printf("wpa_parse_wpa_ie_wpa: ie count botch (pairwise)\
310                             , count %u left %u", count, left);
311                         return -4;
312                 }
313                 for (i = 0; i < count; i++) {
314                         data->pairwise_cipher |= wpa_cypher_to_bitfield(pos);
315                         pos += WPA_SELECTOR_LEN;
316                         left -= WPA_SELECTOR_LEN;
317                 }
318         } else if (left == 1) {
319                 printf("wpa_parse_wpa_ie_wpa: ie too short (for key mgmt)");
320                 return -5;
321         }

323         if (left >= 2) {
324                 data->key_mgmt = 0;
```



```
325                 count = WPA_GET_LE16(pos);
326                 pos += 2;
327                 left -= 2;
328                 if (count == 0 || left < count * WPA_SELECTOR_LEN) {
329                         printf("wpa_parse_wpa_ie_wpa: ie count botch (key mgmt)\
330                             , count %u left %u", count, left);
331                         return -6;
332                 }
333                 for (i = 0; i < count; i++) {
334                         data->key_mgmt |= wpa_key_mgmt_to_bitfield(pos);
335                         pos += WPA_SELECTOR_LEN;
336                         left -= WPA_SELECTOR_LEN;
337                 }
338         } else if (left == 1) {
339                 printf("wpa_parse_wpa_ie_wpa: ie too short (for capabilities)");
340                 return -7;
341         }

343         if (left >= 2) {
344                 data->capabilities = WPA_GET_LE16(pos);
345                 pos += 2;
346                 left -= 2;
347         }

349         if (left > 0) {
350                 printf("wpa_parse_wpa_ie_wpa: ie has %u trailing bytes \
351                     - ignored", left);
352         }

354         return 0;
355 }

357 /**
358  * wpa_parse_wpa_ie_rsn - Parse RSN IE
359  * @rsn_ie: Buffer containing RSN IE
360  * @rsn_ie_len: RSN IE buffer length (including IE number and length octets)
361  * @data: Pointer to structure that will be filled in with parsed data
362  * Returns: 0 on success, <0 on failure
363  */
364 static int wpa_parse_wpa_ie_rsn(const uint8_t *rsn_ie, size_t rsn_ie_len,
365                                 struct wpa_ie_data *data)
366 {
367         const struct rsn_ie_hdr *hdr;
368         const uint8_t *pos;
369         int left;
370         int i, count;

372         memset(data, 0, sizeof(*data));
373         data->proto = WPA_PROTO_RSN;
374         data->pairwise_cipher = WPA_CIPHER_CCMP;
375         data->group_cipher = WPA_CIPHER_CCMP;
376         data->key_mgmt = WPA_KEY_MGMT_IEEE8021X;
377         data->capabilities = 0;
378         data->pmkid = NULL;
379         data->num_pmkid = 0;
380         data->mgmt_group_cipher = 0;

382         if (rsn_ie_len == 0) {
383                 /* No RSN IE - fail silently */
384                 return -1;
385         }

387         if (rsn_ie_len < sizeof(struct rsn_ie_hdr)) {
388                 printf("wpa_parse_wpa_ie_rsn: ie len too short %lu",
389                     (unsigned long) rsn_ie_len);
390                 return -1;
```

```
391         }
393         hdr = (const struct rsn_ie_hdr *) rsn_ie;
395         if (hdr->elem_id != WLAN_EID_RSN ||
396             hdr->len != rsn_ie_len - 2 ||
397             WPA_GET_LE16(hdr->version) != RSN_VERSION) {
398                 printf("wpa_parse_wpa_ie_rsn: malformed ie or unknown version");
399                 return -2;
400         }
402         pos = (const uint8_t *) (hdr + 1);
403         left = rsn_ie_len - sizeof(*hdr);
405         if (left >= RSN_SELECTOR_LEN) {
406                 data->group_cipher = wpa_cypher_to_bitfield(pos);
407                 pos += RSN_SELECTOR_LEN;
408                 left -= RSN_SELECTOR_LEN;
409         } else if (left > 0) {
410                 printf("wpa_parse_wpa_ie_rsn: ie length mismatch, %u too much",
411                     left);
412                 return -3;
413         }
415         if (left >= 2) {
416                 data->pairwise_cipher = 0;
417                 count = WPA_GET_LE16(pos);
418                 pos += 2;
419                 left -= 2;
420                 if (count == 0 || left < count * RSN_SELECTOR_LEN) {
421                         printf("wpa_parse_wpa_ie_rsn: ie count botch (pairwise)\
422                         , count %u left %u", count, left);
423                         return -4;
424                 }
425                 for (i = 0; i < count; i++) {
426                         data->pairwise_cipher |= wpa_cypher_to_bitfield(pos);
427                         pos += RSN_SELECTOR_LEN;
428                         left -= RSN_SELECTOR_LEN;
429                 }
430         } else if (left == 1) {
431                 printf("wpa_parse_wpa_ie_rsn: ie too short (for key mgmt)");
432                 return -5;
433         }
435         if (left >= 2) {
436                 data->key_mgmt = 0;
437                 count = WPA_GET_LE16(pos);
438                 pos += 2;
439                 left -= 2;
440                 if (count == 0 || left < count * RSN_SELECTOR_LEN) {
441                         printf("wpa_parse_wpa_ie_rsn: ie count botch (key mgmt)\
442                         , count %u left %u", count, left);
443                         return -6;
444                 }
445                 for (i = 0; i < count; i++) {
446                         data->key_mgmt |= wpa_key_mgmt_to_bitfield(pos);
447                         pos += RSN_SELECTOR_LEN;
448                         left -= RSN_SELECTOR_LEN;
449                 }
450         } else if (left == 1) {
451                 printf("wpa_parse_wpa_ie_rsn: ie too short (for capabilities)");
452                 return -7;
453         }
455         if (left >= 2) {
456                 data->capabilities = WPA_GET_LE16(pos);
```



```
457                 pos += 2;
458                 left -= 2;
459         }
461         if (left >= 2) {
462                 data->num_pmkid = WPA_GET_LE16(pos);
463                 pos += 2;
464                 left -= 2;
465                 if (left < (int) data->num_pmkid * PMKID_LEN) {
466                         printf("wpa_parse_wpa_ie_rsn: PMKID underflow "
467                                "(num_pmkid=%lu left=%d)",
468                                (unsigned long) data->num_pmkid,
469                                left);
470                         data->num_pmkid = 0;
471                         return -9;
472                 } else {
473                         data->pmkid = pos;
474                         pos += data->num_pmkid * PMKID_LEN;
475                         left -= data->num_pmkid * PMKID_LEN;
476                 }
477         }
479         if (left > 0)
480                 printf("wpa_parse_wpa_ie_rsn: ie has %u trailing bytes \
481                 - ignored", left);
483         return 0;
484 }
486 int wpa_parse_wpa_ie(const uint8_t *wpa_ie, size_t wpa_ie_len,
487                      struct wpa_ie_data *data)
488 {
489         if (wpa_ie_len >= 1 && wpa_ie[0] == WLAN_EID_RSN)
490                 return wpa_parse_wpa_ie_rsn(wpa_ie, wpa_ie_len, data);
491         else
492                 return wpa_parse_wpa_ie_wpa(wpa_ie, wpa_ie_len, data);
493 }
495 /*
496 char * wpa_supplicant_ie_txt(char *pos, char *end, const char *proto,
497                                  const uint8_t *ie, size_t ie_len)
498 {
499         struct wpa_ie_data data;
500         int first, ret;
502         ret = os_snprintf(pos, end - pos, "[%s-", proto);
503         if (ret < 0 || ret >= end - pos)
504                 return pos;
505         pos += ret;
507         if (wpa_parse_wpa_ie(ie, ie_len, &data) < 0) {
508                 ret = os_snprintf(pos, end - pos, "?]");
509                 if (ret < 0 || ret >= end - pos)
510                         return pos;
511                 pos += ret;
512                 return pos;
513         }
515         first = 1;
516         if (data.key_mgmt & WPA_KEY_MGMT_IEEE8021X) {
517                 ret = os_snprintf(pos, end - pos, "%sEAP", first ? "" : "+");
518                 if (ret < 0 || ret >= end - pos)
519                         return pos;
520                 pos += ret;
521                 first = 0;
522         }
```

```
523         if (data.key_mgmt & WPA_KEY_MGMT_PSK) {
524                 ret = os_snprintf(pos, end - pos, "%sPSK", first ? "" : "+");
525                 if (ret < 0 || ret >= end - pos)
526                         return pos;
527                 pos += ret;
528                 first = 0;
529         }
530         if (data.key_mgmt & WPA_KEY_MGMT_WPA_NONE) {
531                 ret = os_snprintf(pos, end - pos, "%sNone", first ? "" : "+");
532                 if (ret < 0 || ret >= end - pos)
533                         return pos;
534                 pos += ret;
535                 first = 0;
536         }

538         pos = wpa_supplicant_cipher_txt(pos, end, data.pairwise_cipher);

540         if (data.capabilities & WPA_CAPABILITY_PREAUTH) {
541                 ret = os_snprintf(pos, end - pos, "-preauth");
542                 if (ret < 0 || ret >= end - pos)
543                         return pos;
544                 pos += ret;
545         }

547         ret = os_snprintf(pos, end - pos, "]");
548         if (ret < 0 || ret >= end - pos)
549                 return pos;
550         pos += ret;

552         return pos;
553 }

555 static const uint8_t * wpa_bss_get_vendor_ie(const struct wpa_bss *bss, uint32_t
556 {
557         const uint8_t *end, *pos;

559         pos = (const uint8_t *) (bss + 1);
560         end = pos + bss->ie_len;

562         while (pos + 1 < end) {
563                 if (pos + 2 + pos[1] > end)
564                         break;
565                 if (pos[0] == WLAN_EID_VENDOR_SPECIFIC && pos[1] >= 4 &&
566                     vendor_type == WPA_GET_BE32(&pos[2]))
567                         return pos;
568                 pos += 2 + pos[1];
569         }

571         return NULL;
572 }

574 static const uint8_t * wpa_bss_get_ie(const struct wpa_bss *bss, uint8_t ie)
575 {
576         const uint8_t *end, *pos;

578         pos = (const uint8_t *) (bss + 1);
579         end = pos + bss->ie_len;

581         while (pos + 1 < end) {
582                 if (pos + 2 + pos[1] > end)
583                         break;
584                 if (pos[0] == ie)
585                         return pos;
586                 pos += 2 + pos[1];
587         }
```



```
589         return NULL;
590 }

592 Format one result on one text line into a buffer.
593 int wpa_supplicant_ctrl_iface_scan_result(
594         struct wpa_supplicant *wpa_s,
595         const struct wpa_bss *bss, char *buf, size_t buflen)
596 {
597         char *pos, *end;
598         int ret;
599         const uint8_t *ie, *ie2;

601         pos = buf;
602         end = buf + buflen;

604         ie = wpa_bss_get_vendor_ie(bss, WPA_IE_VENDOR_TYPE);
605         if (ie)
606                 pos = wpa_supplicant_ie_txt(pos, end, "WPA", ie, 2 + ie[1]);
607         ie2 = wpa_bss_get_ie(bss, WLAN_EID_RSN);
608         if (ie2)
609                 pos = wpa_supplicant_ie_txt(pos, end, "WPA2", ie2, 2 + ie2[1]);
610         pos = wpa_supplicant_wps_ie_txt(wpa_s, pos, end, bss);
611         if (!ie && !ie2 && bss->caps & IEEE80211_CAP_PRIVACY) {
612                 ret = os_snprintf(pos, end - pos, "[WEP]");
613                 if (ret < 0 || ret >= end - pos)
614                         return -1;
615                 pos += ret;
616         }
617         if (bss->caps & IEEE80211_CAP_IBSS) {
618                 ret = os_snprintf(pos, end - pos, "[IBSS]");
619                 if (ret < 0 || ret >= end - pos)
620                         return -1;
621                 pos += ret;
622         }
623         if (bss->caps & IEEE80211_CAP_ESS) {
624                 ret = os_snprintf(pos, end - pos, "[ESS]");
625                 if (ret < 0 || ret >= end - pos)
626                         return -1;
627                 pos += ret;
628         }

630         ret = os_snprintf(pos, end - pos, "\t%s",
631                           wpa_ssid_txt(bss->ssid, bss->ssid_len));
632         if (ret < 0 || ret >= end - pos)
633                 return -1;
634         pos += ret;

636         ret = os_snprintf(pos, end - pos, "\n");
637         if (ret < 0 || ret >= end - pos)
638                 return -1;
639         pos += ret;

641         return pos - buf;
642 }


645 int wpa_supplicant_ctrl_iface_scan_results(
646         struct wpa_supplicant *wpa_s, char *buf, size_t buflen)
647 {
648         char *pos, *end;
649         struct wpa_bss *bss;
650         int ret;

652         pos = buf;
653         end = buf + buflen;
654         ret = os_snprintf(pos, end - pos, "bssid / frequency / signal level / "
```

```
655                             "flags / ssid\n");
656         if (ret < 0 || ret >= end - pos)
657                 return pos - buf;
658         pos += ret;

660         dl_list_for_each(bss, &wpa_s->bss_id, struct wpa_bss, list_id) {
661                 ret = wpa_supplicant_ctrl_iface_scan_result(wpa_s, bss, pos,
662                                                             end - pos);
663                 if (ret < 0 || ret >= end - pos)
664                         return pos - buf;
665                 pos += ret;
666         }

668         return pos - buf;
669 }
670 */
671 #endif /* ! codereview */
```

```
**********************************************************
   20277 Tue Jun 12 19:55:06 2012
new/usr/src/lib/libdladm/common/wpa_if.c
ess_list ioctl now provides all scan results properties for wpa/libdlwlan
first integration of wpa_s control interface client code
first integration of wpa_s wpa_ie parsing code
**********************************************************
```

```
/*
 * Copyright (c) 2002-2012, Jouni Malinen <j@w1.fi>
 *
 * This program is free software; you can redistribute it and/or modify
 * it under the terms of the GNU General Public License version 2 as
 * published by the Free Software Foundation.
 *
 * Alternatively, this software may be distributed under the terms of BSD
 * license.
 */

#include <unistd.h>
#include <stdlib.h>
#include <stdio.h>
#include <strings.h>
#include <errno.h>
#include <ctype.h>
#include <fcntl.h>
#include <sys/socket.h>
#include <libdlwlan.h>

#define WPA_CTRL_RSP "CTRL-RSP-"

enum wpa_cmd_flags {
        cli_cmd_flag_none       = 0x00,
        cli_cmd_flag_sensitive  = 0x01
};

struct wpa_cmd {
        const char *cmd;
        int (*handler)(struct wpa_ctrl *ctrl, int argc, char *argv[]);
        enum wpa_cmd_flags flags;
        const char *usage;
};

/**
 * wpa_ctrl_request - Send a command to wpa_supplicant/hostapd
 * @ctrl: Control interface data from wpa_ctrl_open()
 * @cmd: Command; usually, ASCII text, e.g., "PING"
 * @cmd_len: Length of the cmd in bytes
 * @reply: Buffer for the response
 * @reply_len: Reply buffer length
 * @msg_cb: Callback function for unsolicited messages or %NULL if not used
 * Returns: 0 on success, -1 on error (send or receive failed), -2 on timeout
 *
 * This function is used to send commands to wpa_supplicant/hostapd. Received
 * response will be written to reply and reply_len is set to the actual length
 * of the reply. This function will block for up to two seconds while waiting
 * for the reply. If unsolicited messages are received, the blocking time may
 * be longer.
 *
 * msg_cb can be used to register a callback function that will be called for
 * unsolicited messages received while waiting for the command response. These
 * messages may be received if wpa_ctrl_request() is called at the same time as
 * wpa_supplicant/hostapd is sending such a message. This can happen only if
 * the program has used wpa_ctrl_attach() to register itself as a monitor for
 * event messages. Alternatively to msg_cb, programs can register two control
 * interface connections and use one of them for commands and the other one for
 * receiving event messages, in other words, call wpa_ctrl_attach() only for
```



```
 * the control interface connection that will be used for event messages.
 */
static int wpa_ctrl_request(struct wpa_ctrl *ctrl, const char *cmd,
    size_t cmd_len, char *reply, size_t *reply_len,
    void (*msg_cb)(char *msg, size_t len));

/* These are temporary UNIX sockets created for receiving requests messages*/
#define CTRL_IFACE_CLIENT_DIR "/var/run"
#define CTRL_IFACE_CLIENT_PREFIX "wpa_ctrl_"

struct wpa_ctrl * wpa_ctrl_open(const char *ctrl_path)
{
        struct wpa_ctrl *ctrl;
        static int counter = 0;
        int ret;
        size_t res;
        int tries = 0;

        ctrl = malloc(sizeof(*ctrl));
        if (ctrl == NULL)
                return NULL;
        memset(ctrl, 0, sizeof(*ctrl));

        ctrl->s = socket(PF_UNIX, SOCK_DGRAM, 0);
        if (ctrl->s < 0) {
                free(ctrl);
                return NULL;
        }

        ctrl->local.sun_family = AF_UNIX;
        counter++;
try_again:
        ret = snprintf(ctrl->local.sun_path, sizeof(ctrl->local.sun_path),
            CTRL_IFACE_CLIENT_DIR "/"
            CTRL_IFACE_CLIENT_PREFIX "%d-%d", (int) getpid(), counter);
        if (ret < 0 || (size_t) ret >= sizeof(ctrl->local.sun_path)) {
                close(ctrl->s);
                free(ctrl);
                return NULL;
        }
        tries++;
        if (bind(ctrl->s, (struct sockaddr *) &ctrl->local,
            sizeof(ctrl->local)) < 0) {
                if (errno == EADDRINUSE && tries < 2) {
                        /*
                         * getpid() returns unique identifier for this instance
                         * of wpa_ctrl, so the existing socket file must have
                         * been left by unclean termination of an earlier run.
                         * Remove the file and try again.
                         */
                        unlink(ctrl->local.sun_path);
                        goto try_again;
                }
                close(ctrl->s);
                free(ctrl);
                return NULL;
        }

        ctrl->dest.sun_family = AF_UNIX;
        res = strlcpy(ctrl->dest.sun_path, ctrl_path,
                      sizeof(ctrl->dest.sun_path));
        if (res >= sizeof(ctrl->dest.sun_path)) {
                close(ctrl->s);
                free(ctrl);
                return NULL;
        }
```

```
126         if (connect(ctrl->s, (struct sockaddr *) &ctrl->dest,
127                     sizeof(ctrl->dest)) < 0) {
128                 close(ctrl->s);
129                 unlink(ctrl->local.sun_path);
130                 free(ctrl);
131                 return NULL;
132         }
133
134         return ctrl;
135 }
136
137 void wpa_ctrl_close(struct wpa_ctrl *ctrl)
138 {
139         if (ctrl == NULL)
140                 return;
141         unlink(ctrl->local.sun_path);
142         if (ctrl->s >= 0)
143                 close(ctrl->s);
144         free(ctrl);
145 }
146
147 static int wpa_ctrl_request(struct wpa_ctrl *ctrl, const char *cmd, size_t cmd_l
148                             char *reply, size_t *reply_len,
149                             void (*msg_cb)(char *msg, size_t len))
150 {
151         struct timeval tv;
152         int res;
153         fd_set rfds;
154         const char *_cmd;
155         char *cmd_buf = NULL;
156         size_t _cmd_len;
157
158         {
159                 _cmd = cmd;
160                 _cmd_len = cmd_len;
161         }
162
163         if (send(ctrl->s, _cmd, _cmd_len, 0) < 0) {
164                 free(cmd_buf);
165                 return -1;
166         }
167         free(cmd_buf);
168
169         for (;;) {
170                 tv.tv_sec = 10;
171                 tv.tv_usec = 0;
172                 FD_ZERO(&rfds);
173                 FD_SET(ctrl->s, &rfds);
174                 res = select(ctrl->s + 1, &rfds, NULL, NULL, &tv);
175                 if (res < 0)
176                         return res;
177                 if (FD_ISSET(ctrl->s, &rfds)) {
178                         res = recv(ctrl->s, reply, *reply_len, 0);
179                         if (res < 0)
180                                 return res;
181                         if (res > 0 && reply[0] == '<') {
182                                 /* This is an unsolicited message from
183                                  * wpa_supplicant, not the reply to the
184                                  * request. Use msg_cb to report this to the
185                                  * caller. */
186                                 if (msg_cb) {
187                                         /* Make sure the message is nul
188                                          * terminated. */
189                                         if ((size_t) res == *reply_len)
190                                                 res = (*reply_len) - 1;
191                                         reply[res] = '\0';
```



```
192                                         msg_cb(reply, res);
193                                 }
194                                 continue;
195                         }
196                         *reply_len = res;
197                         break;
198                 } else {
199                         return -2;
200                 }
201         }
202         return 0;
203 }
204
205 static void wpa_msg_cb(char *msg, size_t len)
206 {
207         snprintf(msg, len, "%s\n");
208 }
209
210 static int _wpa_ctrl_command(struct wpa_ctrl *ctrl, char *cmd, int print)
211 {
212         char buf[2048];
213         size_t len;
214         int ret;
215
216         len = sizeof(buf) - 1;
217         ret = wpa_ctrl_request(ctrl, cmd, strlen(cmd), buf, &len,
218                                wpa_msg_cb);
219         if (ret == -2) {
220                 printf("'%s' command timed out.\n", cmd);
221                 return -2;
222         } else if (ret < 0) {
223                 printf("'%s' command failed.\n", cmd);
224                 return -1;
225         }
226         if (print) {
227                 buf[len] = '\0';
228                 printf("%s", buf);
229         }
230         return 0;
231 }
232
233 static int wpa_ctrl_command(struct wpa_ctrl *ctrl, char *cmd)
234 {
235         return _wpa_ctrl_command(ctrl, cmd, 1);
236 }
237
238 /* control interface commands routines */
239
240 static int wpa_cmd_level(struct wpa_ctrl *ctrl, int argc, char *argv[])
241 {
242         char cmd[256];
243         int res;
244
245         if (argc != 1) {
246                 printf("Invalid LEVEL command: needs one argument (debug "
247                        "level)\n");
248                 return -1;
249         }
250         res = snprintf(cmd, sizeof(cmd), "LEVEL %s", argv[0]);
251         if (res < 0 || (size_t) res >= sizeof(cmd) - 1) {
252                 printf("Too long LEVEL command.\n");
253                 return -1;
254         }
255         return wpa_ctrl_command(ctrl, cmd);
256 }
```

```
258 static int wpa_cmd_reassociate(struct wpa_ctrl *ctrl, int argc,
259                                 char *argv[])
260 {
261         return wpa_ctrl_command(ctrl, "REASSOCIATE");
262 }

264 static int wpa_cmd_identity(struct wpa_ctrl *ctrl, int argc, char *argv[])
265 {
266         char cmd[256], *pos, *end;
267         int i, ret;

269         if (argc < 2) {
270                 printf("Invalid IDENTITY command: needs two arguments "
271                        "(network id and identity)\n");
272                 return -1;
273         }

275         end = cmd + sizeof(cmd);
276         pos = cmd;
277         ret = snprintf(pos, end - pos, WPA_CTRL_RSP "IDENTITY-%s:%s",
278                        argv[0], argv[1]);
279         if (ret < 0 || ret >= end - pos) {
280                 printf("Too long IDENTITY command.\n");
281                 return -1;
282         }
283         pos += ret;
284         for (i = 2; i < argc; i++) {
285                 ret = snprintf(pos, end - pos, " %s", argv[i]);
286                 if (ret < 0 || ret >= end - pos) {
287                         printf("Too long IDENTITY command.\n");
288                         return -1;
289                 }
290                 pos += ret;
291         }

293         return wpa_ctrl_command(ctrl, cmd);
294 }

296 static int wpa_cmd_password(struct wpa_ctrl *ctrl, int argc, char *argv[])
297 {
298         char cmd[256], *pos, *end;
299         int i, ret;

301         if (argc < 2) {
302                 printf("Invalid PASSWORD command: needs two arguments "
303                        "(network id and password)\n");
304                 return -1;
305         }

307         end = cmd + sizeof(cmd);
308         pos = cmd;
309         ret = snprintf(pos, end - pos, WPA_CTRL_RSP "PASSWORD-%s:%s",
310                        argv[0], argv[1]);
311         if (ret < 0 || ret >= end - pos) {
312                 printf("Too long PASSWORD command.\n");
313                 return -1;
314         }
315         pos += ret;
316         for (i = 2; i < argc; i++) {
317                 ret = snprintf(pos, end - pos, " %s", argv[i]);
318                 if (ret < 0 || ret >= end - pos) {
319                         printf("Too long PASSWORD command.\n");
320                         return -1;
321                 }
322                 pos += ret;
323         }
```



```
325         return wpa_ctrl_command(ctrl, cmd);
326 }

328 static int wpa_cmd_new_password(struct wpa_ctrl *ctrl, int argc,
329                                 char *argv[])
330 {
331         char cmd[256], *pos, *end;
332         int i, ret;

334         if (argc < 2) {
335                 printf("Invalid NEW_PASSWORD command: needs two arguments "
336                        "(network id and password)\n");
337                 return -1;
338         }

340         end = cmd + sizeof(cmd);
341         pos = cmd;
342         ret = snprintf(pos, end - pos, WPA_CTRL_RSP "NEW_PASSWORD-%s:%s",
343                        argv[0], argv[1]);
344         if (ret < 0 || ret >= end - pos) {
345                 printf("Too long NEW_PASSWORD command.\n");
346                 return -1;
347         }
348         pos += ret;
349         for (i = 2; i < argc; i++) {
350                 ret = snprintf(pos, end - pos, " %s", argv[i]);
351                 if (ret < 0 || ret >= end - pos) {
352                         printf("Too long NEW_PASSWORD command.\n");
353                         return -1;
354                 }
355                 pos += ret;
356         }

358         return wpa_ctrl_command(ctrl, cmd);
359 }


362 static int wpa_cmd_pin(struct wpa_ctrl *ctrl, int argc, char *argv[])
363 {
364         char cmd[256], *pos, *end;
365         int i, ret;

367         if (argc < 2) {
368                 printf("Invalid PIN command: needs two arguments "
369                        "(network id and pin)\n");
370                 return -1;
371         }

373         end = cmd + sizeof(cmd);
374         pos = cmd;
375         ret = snprintf(pos, end - pos, WPA_CTRL_RSP "PIN-%s:%s",
376                        argv[0], argv[1]);
377         if (ret < 0 || ret >= end - pos) {
378                 printf("Too long PIN command.\n");
379                 return -1;
380         }
381         pos += ret;
382         for (i = 2; i < argc; i++) {
383                 ret = snprintf(pos, end - pos, " %s", argv[i]);
384                 if (ret < 0 || ret >= end - pos) {
385                         printf("Too long PIN command.\n");
386                         return -1;
387                 }
388                 pos += ret;
389         }
```

```
390         return wpa_ctrl_command(ctrl, cmd);
391 }

393 static int wpa_cmd_passphrase(struct wpa_ctrl *ctrl, int argc,
394                               char *argv[])
395 {
396         char cmd[256], *pos, *end;
397         int i, ret;

399         if (argc < 2) {
400                 printf("Invalid PASSPHRASE command: needs two arguments "
401                        "(network id and passphrase)\n");
402                 return -1;
403         }

405         end = cmd + sizeof(cmd);
406         pos = cmd;
407         ret = snprintf(pos, end - pos, WPA_CTRL_RSP "PASSPHRASE-%s:%s",
408                        argv[0], argv[1]);
409         if (ret < 0 || ret >= end - pos) {
410                 printf("Too long PASSPHRASE command.\n");
411                 return -1;
412         }
413         pos += ret;
414         for (i = 2; i < argc; i++) {
415                 ret = snprintf(pos, end - pos, " %s", argv[i]);
416                 if (ret < 0 || ret >= end - pos) {
417                         printf("Too long PASSPHRASE command.\n");
418                         return -1;
419                 }
420                 pos += ret;
421         }

423         return wpa_ctrl_command(ctrl, cmd);
424 }


427 static int wpa_cmd_bssid(struct wpa_ctrl *ctrl, int argc, char *argv[])
428 {
429         char cmd[256], *pos, *end;
430         int i, ret;

432         if (argc < 2) {
433                 printf("Invalid BSSID command: needs two arguments (network "
434                        "id and BSSID)\n");
435                 return -1;
436         }

438         end = cmd + sizeof(cmd);
439         pos = cmd;
440         ret = snprintf(pos, end - pos, "BSSID");
441         if (ret < 0 || ret >= end - pos) {
442                 printf("Too long BSSID command.\n");
443                 return -1;
444         }
445         pos += ret;
446         for (i = 0; i < argc; i++) {
447                 ret = snprintf(pos, end - pos, " %s", argv[i]);
448                 if (ret < 0 || ret >= end - pos) {
449                         printf("Too long BSSID command.\n");
450                         return -1;
451                 }
452                 pos += ret;
453         }

455         return wpa_ctrl_command(ctrl, cmd);
```



```
456 }

458 static int wpa_cmd_log_level(struct wpa_ctrl *ctrl, int argc, char *argv[])
459 {
460         char cmd[256], *pos, *end;
461         int i, ret;

463         end = cmd + sizeof(cmd);
464         pos = cmd;
465         ret = snprintf(pos, end - pos, "LOG_LEVEL");
466         if (ret < 0 || ret >= end - pos) {
467                 printf("Too long LOG_LEVEL command.\n");
468                 return -1;
469         }
470         pos += ret;
471         for (i = 0; i < argc; i++) {
472                 ret = snprintf(pos, end - pos, " %s", argv[i]);
473                 if (ret < 0 || ret >= end - pos) {
474                         printf("Too long LOG_LEVEL command.\n");
475                         return -1;
476                 }
477                 pos += ret;
478         }

480         return wpa_ctrl_command(ctrl, cmd);
481 }

483 static int wpa_cmd_select_network(struct wpa_ctrl *ctrl, int argc,
484                                   char *argv[])
485 {
486         char cmd[32];
487         int res;

489         if (argc < 1) {
490                 printf("Invalid SELECT_NETWORK command: needs one argument "
491                        "(network id)\n");
492                 return -1;
493         }

495         res = snprintf(cmd, sizeof(cmd), "SELECT_NETWORK %s", argv[0]);
496         if (res < 0 || (size_t) res >= sizeof(cmd))
497                 return -1;
498         cmd[sizeof(cmd) - 1] = '\0';

500         return wpa_ctrl_command(ctrl, cmd);
501 }


504 static int wpa_cmd_enable_network(struct wpa_ctrl *ctrl, int argc,
505                                   char *argv[])
506 {
507         char cmd[32];
508         int res;

510         if (argc < 1) {
511                 printf("Invalid ENABLE_NETWORK command: needs one argument "
512                        "(network id)\n");
513                 return -1;
514         }

516         res = snprintf(cmd, sizeof(cmd), "ENABLE_NETWORK %s", argv[0]);
517         if (res < 0 || (size_t) res >= sizeof(cmd))
518                 return -1;
519         cmd[sizeof(cmd) - 1] = '\0';

521         return wpa_ctrl_command(ctrl, cmd);
```

```
522 }


525 static int wpa_cmd_disable_network(struct wpa_ctrl *ctrl, int argc,
526                                     char *argv[])
527 {
528         char cmd[32];
529         int res;

531         if (argc < 1) {
532                 printf("Invalid DISABLE_NETWORK command: needs one argument "
533                        "(network id)\n");
534                 return -1;
535         }

537         res = snprintf(cmd, sizeof(cmd), "DISABLE_NETWORK %s", argv[0]);
538         if (res < 0 || (size_t) res >= sizeof(cmd))
539                 return -1;
540         cmd[sizeof(cmd) - 1] = '\0';

542         return wpa_ctrl_command(ctrl, cmd);
543 }


546 static int wpa_cmd_add_network(struct wpa_ctrl *ctrl, int argc,
547                                char *argv[])
548 {
549         return wpa_ctrl_command(ctrl, "ADD_NETWORK");
550 }


553 static int wpa_cmd_remove_network(struct wpa_ctrl *ctrl, int argc,
554                                   char *argv[])
555 {
556         char cmd[32];
557         int res;

559         if (argc < 1) {
560                 printf("Invalid REMOVE_NETWORK command: needs one argument "
561                        "(network id)\n");
562                 return -1;
563         }

565         res = snprintf(cmd, sizeof(cmd), "REMOVE_NETWORK %s", argv[0]);
566         if (res < 0 || (size_t) res >= sizeof(cmd))
567                 return -1;
568         cmd[sizeof(cmd) - 1] = '\0';

570         return wpa_ctrl_command(ctrl, cmd);
571 }

573 static int wpa_cmd_set_network(struct wpa_ctrl *ctrl, int argc,
574                                char *argv[])
575 {
576         char cmd[256];
577         int res;

579         if (argc != 3) {
580                 printf("Invalid SET_NETWORK command: needs three arguments\n"
581                        "(network id, variable name, and value)\n");
582                 return -1;
583         }

585         res = snprintf(cmd, sizeof(cmd), "SET_NETWORK %s %s %s",
586                        argv[0], argv[1], argv[2]);
587         if (res < 0 || (size_t) res >= sizeof(cmd) - 1) {
```



```
588                 printf("Too long SET_NETWORK command.\n");
589                 return -1;
590         }
591         return wpa_ctrl_command(ctrl, cmd);
592 }

594 static int wpa_cmd_disconnect(struct wpa_ctrl *ctrl, int argc,
595                               char *argv[])
596 {
597         return wpa_ctrl_command(ctrl, "DISCONNECT");
598 }

600 static int wpa_cmd_reconnect(struct wpa_ctrl *ctrl, int argc,
601                              char *argv[])
602 {
603         return wpa_ctrl_command(ctrl, "RECONNECT");
604 }

606 static int wpa_cmd_scan(struct wpa_ctrl *ctrl, int argc, char *argv[])
607 {
608         return wpa_ctrl_command(ctrl, "SCAN");
609 }


612 static int wpa_cmd_scan_results(struct wpa_ctrl *ctrl, int argc,
613                                 char *argv[])
614 {
615         return wpa_ctrl_command(ctrl, "SCAN_RESULTS");
616 }

618 static int wpa_cmd_terminate(struct wpa_ctrl *ctrl, int argc,
619                              char *argv[])
620 {
621         return wpa_ctrl_command(ctrl, "TERMINATE");
622 }

624 static int wpa_cmd_interface_add(struct wpa_ctrl *ctrl, int argc,
625                                  char *argv[])
626 {
627         char cmd[256];
628         int res;

630         if (argc < 1) {
631                 printf("Invalid INTERFACE_ADD command: needs at least one "
632                        "argument (interface name)\n"
633                        "All arguments: ifname confname driver ctrl_interface "
634                        "driver_param bridge_name\n");
635                 return -1;
636         }

638         /*
639          * INTERFACE_ADD <ifname>TAB<confname>TAB<driver>TAB<ctrl_interface>TAB
640          * <driver_param>TAB<bridge_name>
641          */
642         res = snprintf(cmd, sizeof(cmd),
643                        "INTERFACE_ADD %s\t%s\t%s\t%s\t%s\t%s",
644                        argv[0],
645                        argc > 1 ? argv[1] : "", argc > 2 ? argv[2] : "",
646                        argc > 3 ? argv[3] : "", argc > 4 ? argv[4] : "",
647                        argc > 5 ? argv[5] : "");
648         if (res < 0 || (size_t) res >= sizeof(cmd))
649                 return -1;
650         cmd[sizeof(cmd) - 1] = '\0';
651         return wpa_ctrl_command(ctrl, cmd);
652 }
```

```
655 static int wpa_cmd_interface_remove(struct wpa_ctrl *ctrl, int argc,
656                                       char *argv[])
657 {
658         char cmd[128];
659         int res;

661         if (argc != 1) {
662                 printf("Invalid INTERFACE_REMOVE command: needs one argument "
663                        "(interface name)\n");
664                 return -1;
665         }

667         res = snprintf(cmd, sizeof(cmd), "INTERFACE_REMOVE %s", argv[0]);
668         if (res < 0 || (size_t) res >= sizeof(cmd))
669                 return -1;
670         cmd[sizeof(cmd) - 1] = '\0';
671         return wpa_ctrl_command(ctrl, cmd);
672 }

674 static int wpa_cmd_ap_scan(struct wpa_ctrl *ctrl, int argc, char *argv[])
675 {
676         char cmd[256];
677         int res;

679         if (argc != 1) {
680                 printf("Invalid AP_SCAN command: needs one argument (ap_scan "
681                        "value)\n");
682                 return -1;
683         }
684         res = snprintf(cmd, sizeof(cmd), "AP_SCAN %s", argv[0]);
685         if (res < 0 || (size_t) res >= sizeof(cmd) - 1) {
686                 printf("Too long AP_SCAN command.\n");
687                 return -1;
688         }
689         return wpa_ctrl_command(ctrl, cmd);
690 }

692 static int wpa_cmd_suspend(struct wpa_ctrl *ctrl, int argc, char *argv[])
693 {
694         return wpa_ctrl_command(ctrl, "SUSPEND");
695 }

697 static struct wpa_cmd wpa_commands[] = {
698         { "level", wpa_cmd_level,
699           cli_cmd_flag_none,
700           "<debug level> = change debug level" },
701         { "reassociate", wpa_cmd_reassociate,
702           cli_cmd_flag_none,
703           "= force reassociation" },
704         { "identity", wpa_cmd_identity,
705           cli_cmd_flag_none,
706           "<network id> <identity> = configure identity for an SSID" },
707         { "password", wpa_cmd_password,
708           cli_cmd_flag_sensitive,
709           "<network id> <password> = configure password for an SSID" },
710         { "new_password", wpa_cmd_new_password,
711           cli_cmd_flag_sensitive,
712           "<network id> <password> = change password for an SSID" },
713         { "pin", wpa_cmd_pin,
714           cli_cmd_flag_sensitive,
715           "<network id> <pin> = configure pin for an SSID" },
716         { "passphrase", wpa_cmd_passphrase,
717           cli_cmd_flag_sensitive,
718           "<network id> <passphrase> = configure private key passphrase\n"
719           "  for an SSID" },
```



```
720         { "bssid", wpa_cmd_bssid,
721           cli_cmd_flag_none,
722           "<network id> <BSSID> = set preferred BSSID for an SSID" },
723         { "log_level", wpa_cmd_log_level,
724           cli_cmd_flag_none,
725           "<level> [<timestamp>] = update the log level/timestamp\n"
726           "log_level = display the current log level and log options" },
727         { "select_network", wpa_cmd_select_network,
728           cli_cmd_flag_none,
729           "<network id> = select a network (disable others)" },
730         { "enable_network", wpa_cmd_enable_network,
731           cli_cmd_flag_none,
732           "<network id> = enable a network" },
733         { "disable_network", wpa_cmd_disable_network,
734           cli_cmd_flag_none,
735           "<network id> = disable a network" },
736         { "add_network", wpa_cmd_add_network,
737           cli_cmd_flag_none,
738           "= add a network" },
739         { "remove_network", wpa_cmd_remove_network,
740           cli_cmd_flag_none,
741           "<network id> = remove a network" },
742         { "set_network", wpa_cmd_set_network,
743           cli_cmd_flag_sensitive,
744           "<network id> <variable> <value> = set network variables (shows\n"
745           "  list of variables when run without arguments)" },
746         { "disconnect", wpa_cmd_disconnect,
747           cli_cmd_flag_none,
748           "= disconnect and wait for reassociate/reconnect command before\n"
749           "  connecting" },
750         { "reconnect", wpa_cmd_reconnect,
751           cli_cmd_flag_none,
752           "= like reassociate, but only takes effect if already disconnected"
753         },
754         { "scan", wpa_cmd_scan,
755           cli_cmd_flag_none,
756           "= request new BSS scan" },
757         { "scan_results", wpa_cmd_scan_results,
758           cli_cmd_flag_none,
759           "= get latest scan results" },
760         { "terminate", wpa_cmd_terminate,
761           cli_cmd_flag_none,
762           "= terminate wpa_supplicant" },
763         { "interface_add", wpa_cmd_interface_add,
764           cli_cmd_flag_none,
765           "<ifname> <confname> <driver> <ctrl_interface> <driver_param>\n"
766           "  <bridge_name> = adds new interface, all parameters but <ifname>\n"
767           "  are optional" },
768         { "interface_remove", wpa_cmd_interface_remove,
769           cli_cmd_flag_none,
770           "<ifname> = removes the interface" },
771         { "ap_scan", wpa_cmd_ap_scan,
772           cli_cmd_flag_none,
773           "<value> = set ap_scan parameter" },
774         { "suspend", wpa_cmd_suspend, cli_cmd_flag_none,
775           "= notification of suspend/hibernate" },
776         { NULL, NULL, cli_cmd_flag_none, NULL }
777 };

779 int wpa_request(struct wpa_ctrl *ctrl, int argc, char *argv[])
780 {
781         struct wpa_cmd *cmd, *match = NULL;
782         int count;
783         int ret = 0;

785         count = 0;
```

```
786         cmd = wpa_commands;
787         while (cmd->cmd) {
788                 if (strncasecmp(cmd->cmd, argv[0], strlen(argv[0])) == 0)
789                 {
790                         match = cmd;
791                         if (strcasecmp(cmd->cmd, argv[0]) == 0) {
792                                 /* we have an exact match */
793                                 count = 1;
794                                 break;
795                         }
796                         count++;
797                 }
798                 cmd++;
799         }

801         if (count > 1) {
802                 printf("Ambiguous command '%s'; possible commands:", argv[0]);
803                 cmd = wpa_commands;
804                 while (cmd->cmd) {
805                         if (strncasecmp(cmd->cmd, argv[0],
806                                         strlen(argv[0])) == 0) {
807                                 printf(" %s", cmd->cmd);
808                         }
809                         cmd++;
810                 }
811                 printf("\n");
812                 ret = 1;
813         } else if (count == 0) {
814                 printf("Unknown command '%s'\n", argv[0]);
815                 ret = 1;
816         } else {
817                 ret = match->handler(ctrl, argc - 1, &argv[1]);
818         }

820         return ret;
821 }
822 #endif /* ! codereview */
```

```
**********************************************************
   23059 Tue Jun 12 19:55:06 2012
new/usr/src/lib/libnwam/common/libnwam_known_wlan.c
Implemented ctrl_if network block configuration routine
**********************************************************
______unchanged_portion_omitted_

 523 static nwam_error_t
 524 valid_secmode(nwam_value_t value)
 525 {
 526         uint64_t secmode;

 528         if (nwam_value_get_uint64(value, &secmode) != NWAM_SUCCESS)
 529                 return (NWAM_ENTITY_INVALID_VALUE);

 531         if (secmode != DLADM_WLAN_SECMODE_NONE &&
 532             secmode != DLADM_WLAN_SECMODE_WEP &&
 533             secmode != DLADM_WLAN_SECMODE_PSK &&
 534             secmode != DLADM_WLAN_SECMODE_EAP)
 533             secmode != DLADM_WLAN_SECMODE_WPA)
 535                 return (NWAM_ENTITY_INVALID_VALUE);

 537         return (NWAM_SUCCESS);
 538 }
______unchanged_portion_omitted_
```

****
30961 Tue Jun 12 19:55:07 2012
new/usr/src/lib/libnwam/common/libnwam_values.c
Implemented ctrl_if network block configuration routine
****
_____***unchanged_portion_omitted_***

```
 722 struct nwam_value_entry known_wlan_prop_security_mode_entries[] =
 723 {
 724         { "none", DLADM_WLAN_SECMODE_NONE },
 725         { "wep", DLADM_WLAN_SECMODE_WEP },
 726         { "wpa-psk", DLADM_WLAN_SECMODE_PSK },
 727         { "wpa-eap", DLADM_WLAN_SECMODE_EAP },
 726         { "wpa", DLADM_WLAN_SECMODE_WPA },
 728         { NULL, 0 }
 729 };
```

_____***unchanged_portion_omitted_***

```
**********************************************************
   14198 Tue Jun 12 19:55:09 2012
new/usr/src/lib/libsecdb/auth_attr.txt
removed wificonfig tool
are /dev/wifi/* devices links now deprecated?
**********************************************************
```

```
#
# 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) 1999, 2010, Oracle and/or its affiliates. All rights reserved.


#
# /etc/security/auth_attr
#
# authorizations. see auth_attr(4)
#
solaris.:::All Solaris Authorizations::help=AllSolAuthsHeader.html
solaris.grant:::Grant All Solaris Authorizations::help=PriAdmin.html
#
solaris.admin.idmap.rules:::Manage Identity Mapping Rules::help=IdmapRules.html
#
solaris.admin.wusb.:::Administer Wireless USB::help=WUSBHeader.html
solaris.admin.wusb.read:::Read Wireless USB Host and Device Information::help=WU
solaris.admin.wusb.modify:::Add or delete information of Wireless USB Device::he
solaris.admin.wusb.host:::Manage Wireless USB Host::help=WUSBhost.html
#
solaris.audit.:::Audit System-wide Management::help=AuditHeader.html
#
solaris.device.:::Device Allocation::help=DevAllocHeader.html
solaris.device.allocate:::Allocate Device::help=DevAllocate.html
solaris.device.config:::Configure Device Attributes::help=DevConfig.html
solaris.device.grant:::Delegate Device Administration::help=DevGrant.html
solaris.device.revoke:::Revoke or Reclaim Device::help=DevRevoke.html
solaris.device.cdrw:::CD-R/RW Recording Authorizations::help=DevCDRW.html
solaris.device.mount.:::Device Mount::help=DevMount.html
solaris.device.mount.alloptions.fixed:::Device Mount Fixed With All Options::hel
solaris.device.mount.alloptions.removable:::Device Mount Removable With All Opti
solaris.device.mount.fixed:::Device Mount Fixed::help=DevMount.html
solaris.device.mount.removable:::Device Mount Removable::help=DevMount.html
#
solaris.dhcpmgr.:::DHCP Service Management::help=DhcpmgrHeader.html
solaris.dhcpmgr.write:::Modify DHCP Service Configuration::help=DhcpmgrWrite.htm
#
solaris.file.:::File Operations::help=FileHeader.html
solaris.file.chown:::Change File Owner::help=FileChown.html
solaris.file.owner:::Act as File Owner::help=FileOwner.html
#
```



```
solaris.hotplug.:::Hotplug::help=HotplugHeader.html
solaris.hotplug.modify:::Modify Hotplug Connections::help=HotplugModify.html
#
solaris.jobs.:::Job Scheduler::help=JobHeader.html
solaris.jobs.admin:::Manage All Jobs::help=AuthJobsAdmin.html
solaris.jobs.grant:::Delegate Cron & At Administration::help=JobsGrant.html
solaris.jobs.user:::Manage Owned Jobs::help=AuthJobsUser.html
#
solaris.label.:::Label Management::help=LabelHeader.html
solaris.label.file.downgrade:::Downgrade File Label::help=LabelFileDowngrade.htm
solaris.label.file.upgrade:::Upgrade File Label::help=LabelFileUpgrade.html
solaris.label.print:::View Printer Queue at All Labels::help=LabelPrint.html
solaris.label.range:::Set Label Outside User Accred Range::help=LabelRange.html
solaris.label.win.downgrade:::Downgrade DragNDrop or CutPaste Info::help=LabelWi
solaris.label.win.noview:::DragNDrop or CutPaste without viewing contents::help=
solaris.label.win.upgrade:::Upgrade DragNDrop or CutPaste Info::help=LabelWinUpg
#
solaris.login.:::Login Control::help=LoginHeader.html
solaris.login.enable:::Enable Logins::help=LoginEnable.html
solaris.login.remote:::Remote Login::help=LoginRemote.html
#
solaris.mail.:::Mail::help=MailHeader.html
solaris.mail.mailq:::Mail Queue::help=MailQueue.html
#
solaris.network.:::Network::help=NetworkHeader.html
solaris.network.autoconf.read:::View Network Auto-Magic Config::help=NetworkAuto
solaris.network.autoconf.select:::Enable/Disable Network Auto-Magic Config::help
solaris.network.autoconf.wlan:::Create Network Auto-Magic Config for Known WLANs
solaris.network.autoconf.write:::Create Network Auto-Magic Config::help=NetworkA
solaris.network.ilb.config:::Network ILB Configuration::help=NetworkILBconf.html
solaris.network.ilb.enable:::Network ILB Enable Configuration::help=NetworkILBen
solaris.network.interface.config:::Network Interface Configuration::help=Network
solaris.network.link.security:::Link Security::help=LinkSecurity.html
solaris.network.wifi.config:::Wifi Config::help=WifiConfig.html
solaris.network.wifi.wep:::Wifi Wep::help=WifiWep.html
solaris.network.vrrp:::Administer VRRP::help=NetworkVRRP.html
#
solaris.print.:::Printer Management::help=PrintHeader.html
solaris.print.admin:::Administer Printer::help=PrintAdmin.html
solaris.print.cancel:::Cancel Print Job::help=PrintCancel.html
solaris.print.list:::List Jobs in Printer Queue::help=PrintList.html
solaris.print.nobanner:::Print without Banner::help=PrintNoBanner.html
solaris.print.ps:::Print Postscript::help=PrintPs.html
solaris.print.unlabeled:::Print without Label::help=PrintUnlabeled.html
#
solaris.profmgr.:::Rights::help=ProfmgrHeader.html
solaris.profmgr.assign:::Assign All Rights::help=AuthProfmgrAssign.html
solaris.profmgr.delegate:::Assign Owned Rights::help=AuthProfmgrDelegate.html
solaris.profmgr.write:::Manage Rights::help=AuthProfmgrWrite.html
solaris.profmgr.read:::View Rights::help=AuthProfmgrRead.html
solaris.profmgr.execattr.write:::Manage Commands::help=AuthProfmgrExecattrWrite.
#
solaris.role.:::Roles::help=RoleHeader.html
solaris.role.assign:::Assign All Roles::help=AuthRoleAssign.html
solaris.role.delegate:::Assign Owned Roles::help=AuthRoleDelegate.html
solaris.role.write:::Manage Roles::help=AuthRoleWrite.html
#
solaris.smf.:::SMF Management::help=SmfHeader.html
solaris.smf.modify.:::Modify All SMF Service Properties::help=SmfModifyHeader.ht
solaris.smf.modify.method:::Modify Service Methods::help=SmfModifyMethod.html
solaris.smf.modify.dependency:::Modify Service Dependencies::help=SmfModifyDepen
solaris.smf.modify.application:::Modify Application Type Properties::help=SmfMod
solaris.smf.modify.framework:::Modify Framework Type Properties::help=SmfModifyF
solaris.smf.manage.:::Manage All SMF Service States::help=SmfManageHeader.html
solaris.smf.manage.allocate:::Manage Device Allocation Service::help=SmfAllocate
solaris.smf.manage.audit:::Manage Audit Service States::help=SmfManageAudit.html
```

```
125 solaris.smf.manage.autofs:::Manage Automount Service States::help=SmfAutofsState
126 solaris.smf.manage.bind:::Manage DNS Service States::help=BindStates.html
127 solaris.smf.manage.coreadm:::Manage Coreadm Service States::help=SmfCoreadmState
128 solaris.smf.manage.cron:::Manage Cron Service States::help=SmfCronStates.html
129 solaris.smf.manage.discovery.printers.snmp:::Manage Network Attached Device Disc
130 solaris.smf.manage.extended-accounting.flow:::Manage Flow Extended Accounting Se
131 solaris.smf.manage.extended-accounting.process:::Manage Process Extended Account
132 solaris.smf.manage.extended-accounting.flow:::Manage Task Extended Accounting Se
133 solaris.smf.manage.hal:::Manage HAL Service States::help=SmfHALStates.html
134 solaris.smf.manage.hotplug:::Manage Hotplug Service::help=SmfManageHotplug.html
135 solaris.smf.manage.idmap:::Manage Identity Mapping Service States::help=SmfIdmap
136 solaris.smf.manage.ilb:::Manage Integrated Load Balancer Service States::help=Sm
137 solaris.smf.manage.inetd:::Manage inetd and inetd managed services States::help=
138 solaris.smf.manage.ipsec:::Manage IPsec Service States::help=SmfIPsecStates.html
139 solaris.smf.manage.labels:::Manage label server::help=LabelServer.html
140 solaris.smf.manage.location:::Manage Network Location Service States::help=SmfLo
141 solaris.smf.manage.mdns:::Manage Multicast DNS Service States::help=SmfMDNSState
142 solaris.smf.manage.name-service-cache:::Manage Name Service Cache Daemon Service
143 solaris.smf.manage.nwam:::Manage Network Auto-Magic Service States::help=SmfNWAM
144 solaris.smf.manage.power:::Manage Power Management Service States::help=SmfPower
145 solaris.smf.manage.smb:::Manage SMB Service States::help=SmfSMBStates.html
146 solaris.smf.manage.smbfs:::Manage SMB Client States::help=SmfSMBFSStates.html
147 solaris.smf.manage.reparse:::Manage Reparse Service States::help=SmfReparseState
148 solaris.smf.manage.rmvolmgr:::Manage Rmvolmgr Service States::help=SmfRmvolmgrSt
149 solaris.smf.manage.routing:::Manage Routing Service States::help=SmfRoutingState
150 solaris.smf.manage.rpc.bind:::Manage RPC Program number mapper::help=SmfRPCBind.
151 solaris.smf.manage.sendmail:::Manage Sendmail Service States::help=SmfSendmailSt
152 solaris.smf.manage.smtp-notify:::Manage Email Event Notification Agent::
153 solaris.smf.manage.snmp-notify:::Manage SNMP Event Notification Agent::
154 solaris.smf.manage.ssh:::Manage Secure Shell Service States::help=SmfSshStates.h
155 solaris.smf.manage.stmf:::Manage STMF Service States::help=SmfSTMFStates.html
156 solaris.smf.manage.system-log:::Manage Syslog Service States::help=SmfSyslogStat
157 solaris.smf.manage.tnctl:::Manage Refresh of Trusted Network Parameters::help=TN
158 solaris.smf.manage.tnd:::Manage Trusted Network Daemon::help=TNDaemon.html
159 solaris.smf.manage.vrrp:::Manage VRRP Service States::help=SmfVRRPStates.html
160 solaris.smf.manage.vscan:::Manage VSCAN Service States::help=SmfVscanStates.html
161 solaris.smf.manage.vt:::Manage Virtual Console Service States::help=SmfVtStates.
162 solaris.smf.manage.wpa:::Manage WPA Service States::help=SmfWpaStates.html
163 solaris.smf.manage.ndmp:::Manage NDMP Service States::help=SmfNDMPStates.html
164 solaris.smf.value.:::Change Values of SMF Service Properties::help=SmfValueHeade
165 solaris.smf.value.audit:::Configure the Audit Service::help=SmfValueAudit.html
166 solaris.smf.value.coreadm:::Change Values of SMF Coreadm Properties::help=SmfVal
167 solaris.smf.value.discovery.printers.snmp:::Manage Network Attached Device Disco
168 solaris.smf.value.extended-accounting.flow:::Change Values of Flow Extended Acco
169 solaris.smf.value.extended-accounting.process:::Change Values of Process Extende
170 solaris.smf.value.extended-accounting.task:::Change Values of Task Extended Acco
171 solaris.smf.value.firewall.config::Change Service Firewall Config::help=SmfValu
172 solaris.smf.value.idmap:::Change Values of SMF Identity Mapping Service Properti
173 solaris.smf.value.inetd:::Change values of SMF Inetd configuration paramaters::h
174 solaris.smf.value.ipsec:::Change Values of SMF IPsec Properties::help=SmfValueIP
175 solaris.smf.value.mdns:::Change Values of MDNS Service Properties::help=SmfValue
176 solaris.smf.value.nwam:::Change Values of SMF Network Auto-Magic Properties::hel
177 solaris.smf.value.smb:::Change Values of SMB Service Properties::help=SmfValueSM
178 solaris.smf.read.smb:::Read permission for protected SMF SMB Service Properties:
179 solaris.smf.value.smtp-notify:::Change values of Email Event Notification Agent
180 solaris.smf.value.snmp-notify:::Change values of SNMP Event Notification Agent p
181 solaris.smf.read.stmf:::Read STMF Provider Private Data::help=SmfSTMFRead.html
182 solaris.smf.value.routing:::Change Values of SMF Routing Properties::help=SmfVal
183 solaris.smf.value.tnd:::Change Trusted Network Daemon Service Property Values::h
184 solaris.smf.value.vscan::Change Values of VSCAN Properties::help=SmfValueVscan.
185 solaris.smf.value.vt:::Change Values of Virtual Console Service Properties::help
186 solaris.smf.value.ndmp:::Change Values of SMF NDMP Service Properties::help=SmfV
187 solaris.smf.read.ndmp:::Read permission for protected SMF NDMP Service Propertie
188 #
189 solaris.system.:::Machine Administration::help=SysHeader.html
190 solaris.system.date:::Set Date & Time::help=SysDate.html
```



```
191 solaris.system.maintenance:::Enter Maintenance (single-user) Mode::help=SysMaint
192 solaris.system.shutdown:::Shutdown the System::help=SysShutdown.html
193 solaris.system.power.:::System Power Management::help=SysPowerMgmtHeader.html
194 solaris.system.power.suspend.:::Suspend the System::help=SysPowerMgmtSuspend.htm
195 solaris.system.power.suspend.disk:::Suspend to Disk::help=SysPowerMgmtSuspendtoD
196 solaris.system.power.suspend.ram:::Suspend to RAM::help=SysPowerMgmtSuspendToRAM
197 solaris.system.power.brightness:::Control LCD Brightness::help=SysPowerMgmtBrigh
198 solaris.system.power.cpu:::Manage CPU related power::help=SysCpuPowerMgmt.html
199 solaris.system.sysevent.read:::Retrieve Sysevents::help=SysSyseventRead.html
200 solaris.system.sysevent.write:::Publish Sysevents::help=SysSyseventWrite.html
201 #
202 solaris.smf.modify.stmf:::Modify STMF Properties::help=SmfSTMFValue.html
203 #
204 solaris.smf.manage.isns:::Manage iSNS Service States::help=isnsStates.html
205 solaris.smf.value.isns:::Modify iSNS Service Property Values::help=isnsValue.htm
206 solaris.isnsmgr.write:::Modify iSNS configuration::help=AuthISNSmgrWrite.html
207 solaris.smf.manage.wusb:::Manage Wireless USB Service::help=SmfWusbStates.html
208 solaris.zone.:::Zone Management::help=ZoneHeader.html
209 solaris.zone.clonefrom:::Clone another Zone::help=ZoneCloneFrom.html
210 solaris.zone.login:::Zone Login::help=ZoneLogin.html
211 solaris.zone.manage:::Zone Deployment::help=ZoneManage.html
```

```
**********************************************************
    4724 Tue Jun 12 19:55:10 2012
new/usr/src/lib/libsecdb/help/auths/Makefile
removed wificonfig tool
are /dev/wifi/* devices links now deprecated?
**********************************************************
#
# 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) 1999, 2010, Oracle and/or its affiliates. All rights reserved.

#
# lib/libsecdb/help/auths/Makefile
#

include ../../../../Makefile.master

HTMLENTS = \
        AuditHeader.html \
        DevAllocHeader.html \
        DevAllocate.html \
        DevConfig.html \
        DevCDRW.html \
        DevGrant.html \
        DevRevoke.html \
        HotplugHeader.html \
        HotplugModify.html \
        JobHeader.html \
        AuthJobsAdmin.html \
        JobsGrant.html \
        AuthJobsUser.html \
        LoginEnable.html \
        LoginHeader.html \
        LoginRemote.html \
        MailHeader.html \
        MailQueue.html \
        PriAdmin.html \
        AuthProfmgrAssign.html \
        AuthProfmgrDelegate.html \
        AuthProfmgrExecattrWrite.html \
        AuthProfmgrRead.html \
        ProfmgrHeader.html \
        AuthProfmgrWrite.html \
        AuthRoleAssign.html \
        AuthRoleDelegate.html \
        RoleHeader.html \
        AuthRoleWrite.html \
        SysDate.html \
```



```
        SysHeader.html \
        SysShutdown.html \
        AllSolAuthsHeader.html \
        SysMaintenance.html \
        DhcpmgrHeader.html \
        DhcpmgrWrite.html \
        BindStates.html \
        SmfAllocate.html \
        SmfAutofsStates.html \
        SmfCoreadmStates.html \
        SmfCronStates.html \
        SmfExAcctFlowStates.html \
        SmfExAcctProcessStates.html \
        SmfExAcctTaskStates.html \
        SmfExAcctNetStates.html \
        SmfHeader.html \
        SmfILBStates.html \
        SmfInetdStates.html \
        SmfIPsecStates.html \
        SmfLocationStates.html \
        SmfManageAudit.html \
        SmfManageHeader.html \
        SmfManageHotplug.html \
        SmfMDNSStates.html \
        SmfModifyAppl.html \
        SmfModifyDepend.html \
        SmfModifyFramework.html \
        SmfModifyHeader.html \
        SmfModifyMethod.html \
        SmfNscdStates.html \
        SmfNADDStates.html \
        SmfNDMPStates.html \
        SmfNWAMStates.html \
        SmfPowerStates.html \
        SmfReparseStates.html \
        SmfRoutingStates.html \
        SmfSendmailStates.html \
        SmfSshStates.html \
        SmfSyslogStates.html \
        SmfValueAudit.html \
        SmfValueCoreadm.html \
        SmfValueExAcctFlow.html \
        SmfValueExAcctProcess.html \
        SmfValueExAcctTask.html \
        SmfValueExAcctNet.html \
        SmfValueFirewall.html \
        SmfVtStates.html \
        SmfValueHeader.html \
        SmfValueInetd.html \
        SmfValueIPsec.html \
        SmfValueMDNS.html \
        SmfValueNADD.html \
        SmfValueNDMP.html \
        AuthReadNDMP.html \
        SmfValueNWAM.html \
        SmfValueRouting.html \
        SmfValueSMB.html \
        AuthReadSMB.html \
        SmfSMBFSStates.html \
        SmfSMBStates.html \
        SmfValueVscan.html \
        SmfVscanStates.html \
        SmfValueVt.html \
        SmfVRRPStates.html \
        SmfWpaStates.html \
        NetworkAutoconfRead.html \
```

```
127         NetworkAutoconfSelect.html \
128         NetworkAutoconfWlan.html \
129         NetworkAutoconfWrite.html \
130         NetworkILBconf.html \
131         NetworkILBenable.html \
132         NetworkHeader.html \
133         NetworkVRRP.html \
134         NetworkInterfaceConfig.html \
135         WifiConfig.html \
136         WifiWep.html \
135         LinkSecurity.html \
136         IdmapRules.html \
137         SmfIdmapStates.html \
138         SmfValueIdmap.html \
139         FileChown.html \
140         FileHeader.html \
141         FileOwner.html \
142         LabelFileDowngrade.html \
143         LabelFileUpgrade.html \
144         LabelHeader.html \
145         LabelPrint.html \
146         LabelRange.html \
147         LabelServer.html \
148         LabelWinDowngrade.html \
149         LabelWinNoView.html \
150         LabelWinUpgrade.html \
151         PrintAdmin.html \
152         PrintCancel.html \
153         PrintHeader.html \
154         PrintList.html \
155         PrintNoBanner.html \
156         PrintPs.html \
157         PrintUnlabeled.html \
158         TNDaemon.html \
159         TNctl.html \
160         ValueTND.html \
161         SysPowerMgmtHeader.html \
162         SysPowerMgmtSuspend.html \
163         SysPowerMgmtSuspendtoDisk.html \
164         SysPowerMgmtSuspendtoRAM.html \
165         SysPowerMgmtBrightness.html \
166         SysCpuPowerMgmt.html \
167         SysSyseventRead.html \
168         SysSyseventWrite.html \
169         SmfManageZFSSnap.html \
170         ZoneCloneFrom.html \
171         ZoneHeader.html \
172         ZoneLogin.html \
173         ZoneManage.html

175 HELPDIR=$(ROOT)/usr/lib/help
176 AUTHDIR=$(HELPDIR)/auths
177 LOCALEDIR=$(AUTHDIR)/locale
178 CDIR=$(LOCALEDIR)/C
179 DIRS=$(HELPDIR) $(AUTHDIR) $(LOCALEDIR) $(CDIR)
180 HELPFILES=$(HTMLENTS:%=$(CDIR)/%)

182 MSGDIR=         $(LOCALEDIR)
183 MSGDIRS =       $(HELPDIR) $(AUTHDIR) $(LOCALEDIR)

185 MSGFILES=       $(HTMLENTS)
186 MSGS=           $(MSGFILES:%=$(MSGDIR)/%)

188 FILEMODE= 0444

190 .KEEP_STATE:
```



```
192 all:    $(HTMLENTS)

194 install:        all $(DIRS) $(HELPFILES)

196 _msg: $(MSGDIRS) $(MSGS)

198 $(CDIR)/%: %
199         $(INS.file)

201 $(DIRS):
202         $(INS.dir)

204 $(MSGDIR)/%: %
205         $(INS.file)

207 clean clobber lint:
```

```
**********************************************************
   89941 Tue Jun 12 19:55:11 2012
new/usr/src/pkg/manifests/SUNWcs.mf
removed wificonfig tool
are /dev/wifi/* devices links now deprecated?
**********************************************************
```

```
   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 2011 Nexenta Systems, Inc. All rights reserved.
  25 #

  27 <include SUNWcs.man1.inc>
  28 <include SUNWcs.man1m.inc>
  29 <include SUNWcs.man4.inc>
  30 <include SUNWcs.man5.inc>
  31 <include SUNWcs.man7d.inc>
  32 <include SUNWcs.man7fs.inc>
  33 set name=pkg.fmri value=pkg:/SUNWcs@$(PKGVERS)
  34 set name=pkg.description \
  35     value="core software for a specific instruction-set architecture"
  36 set name=pkg.summary value="Core Solaris"
  37 set name=info.classification value=org.opensolaris.category.2008:System/Core
  38 set name=variant.arch value=$(ARCH)
  39 dir path=dev group=sys
  40 dir path=etc group=sys
  41 dir path=etc/certs group=sys
  42 dir path=etc/cron.d group=sys
  43 dir path=etc/crypto group=sys
  44 dir path=etc/crypto/certs group=sys
  45 dir path=etc/crypto/crls group=sys
  46 dir path=etc/default group=sys
  47 dir path=etc/dev group=sys
  48 dir path=etc/devices group=sys
  49 dir path=etc/dfs group=sys
  50 dir path=etc/dhcp group=sys
  51 dir path=etc/fs group=sys
  52 dir path=etc/fs/dev group=sys
  53 dir path=etc/fs/hsfs group=sys
  54 dir path=etc/fs/ufs group=sys
  55 dir path=etc/ftpd group=sys
  56 dir path=etc/inet group=sys
  57 dir path=etc/init.d group=sys
  58 dir path=etc/lib group=sys
  59 dir path=etc/logadm.d group=sys
  60 dir path=etc/mail group=mail
```



```
  61 dir path=etc/net group=sys
  62 dir path=etc/net/ticlts group=sys
  63 dir path=etc/net/ticots group=sys
  64 dir path=etc/net/ticotsord group=sys
  65 dir path=etc/opt group=sys
  66 dir path=etc/rc0.d group=sys
  67 dir path=etc/rc1.d group=sys
  68 dir path=etc/rc2.d group=sys
  69 dir path=etc/rc3.d group=sys
  70 dir path=etc/rcS.d group=sys
  71 dir path=etc/rpcsec group=sys
  72 dir path=etc/saf
  73 dir path=etc/saf/zsmon group=sys
  74 dir path=etc/sasl group=sys
  75 dir path=etc/security group=sys
  76 dir path=etc/security/audit group=sys
  77 dir path=etc/security/audit/localhost group=sys
  78 dir path=etc/security/auth_attr.d group=sys
  79 dir path=etc/security/dev group=sys
  80 dir path=etc/security/exec_attr.d group=sys
  81 dir path=etc/security/lib group=sys
  82 dir path=etc/security/prof_attr.d group=sys
  83 dir path=etc/skel group=sys
  84 dir path=etc/svc group=sys
  85 dir path=etc/svc/profile group=sys
  86 dir path=etc/svc/profile/site group=sys
  87 dir path=etc/svc/volatile group=sys
  88 dir path=etc/sysevent group=sys
  89 dir path=etc/sysevent/config group=sys
  90 dir path=etc/tm group=sys
  91 dir path=etc/user_attr.d group=sys
  92 dir path=export group=sys
  93 dir path=home group=root mode=0555
  94 dir path=lib
  95 dir path=lib/crypto
  96 dir path=lib/inet
  97 dir path=lib/svc
  98 dir path=lib/svc/bin
  99 dir path=lib/svc/capture
 100 dir path=lib/svc/manifest group=sys
 101 dir path=lib/svc/manifest/application group=sys
 102 dir path=lib/svc/manifest/application/management group=sys
 103 dir path=lib/svc/manifest/application/security group=sys
 104 dir path=lib/svc/manifest/device group=sys
 105 dir path=lib/svc/manifest/milestone group=sys
 106 dir path=lib/svc/manifest/network group=sys
 107 dir path=lib/svc/manifest/network/dns group=sys
 108 dir path=lib/svc/manifest/network/ipsec group=sys
 109 dir path=lib/svc/manifest/network/ldap group=sys
 110 dir path=lib/svc/manifest/network/routing group=sys
 111 dir path=lib/svc/manifest/network/rpc group=sys
 112 dir path=lib/svc/manifest/network/shares group=sys
 113 dir path=lib/svc/manifest/network/ssl group=sys
 114 dir path=lib/svc/manifest/platform group=sys
 115 $(sparc_ONLY)dir path=lib/svc/manifest/platform/sun4u group=sys
 116 dir path=lib/svc/manifest/site group=sys
 117 dir path=lib/svc/manifest/system group=sys
 118 dir path=lib/svc/manifest/system/device group=sys
 119 dir path=lib/svc/manifest/system/filesystem group=sys
 120 dir path=lib/svc/manifest/system/security group=sys
 121 dir path=lib/svc/manifest/system/svc group=sys
 122 dir path=lib/svc/method
 123 dir path=lib/svc/monitor
 124 dir path=lib/svc/seed
 125 dir path=lib/svc/share
 126 dir path=mnt group=sys
```

```
127 dir path=opt group=sys
128 dir path=proc group=root mode=0555
129 dir path=root group=root mode=0700
130 dir path=sbin group=sys
131 dir path=system group=root
132 dir path=system/contract group=root mode=0555
133 dir path=system/object group=root mode=0555
134 dir path=tmp group=sys mode=1777
135 dir path=usr group=sys
136 dir path=usr/bin
137 dir path=usr/bin/$(ARCH32)
138 dir path=usr/bin/$(ARCH64)
139 dir path=usr/ccs
140 dir path=usr/ccs/bin
141 dir path=usr/demo
142 dir path=usr/games
143 dir path=usr/has
144 dir path=usr/has/bin
145 dir path=usr/has/lib
146 dir path=usr/has/man
147 dir path=usr/has/man/man1has
148 dir path=usr/kernel group=sys
149 dir path=usr/kernel/drv group=sys
150 dir path=usr/kernel/drv/$(ARCH64) group=sys
151 dir path=usr/kernel/exec group=sys
152 dir path=usr/kernel/exec/$(ARCH64) group=sys
153 dir path=usr/kernel/fs group=sys
154 dir path=usr/kernel/fs/$(ARCH64) group=sys
155 dir path=usr/kernel/pcbe group=sys
156 dir path=usr/kernel/pcbe/$(ARCH64) group=sys
157 dir path=usr/kernel/sched group=sys
158 dir path=usr/kernel/sched/$(ARCH64) group=sys
159 dir path=usr/kernel/strmod group=sys
160 dir path=usr/kernel/strmod/$(ARCH64) group=sys
161 dir path=usr/kernel/sys group=sys
162 dir path=usr/kernel/sys/$(ARCH64) group=sys
163 dir path=usr/kvm
164 dir path=usr/lib
165 dir path=usr/lib/$(ARCH64)
166 dir path=usr/lib/audit
167 dir path=usr/lib/class
168 dir path=usr/lib/class/FX
169 dir path=usr/lib/class/IA
170 dir path=usr/lib/class/RT
171 dir path=usr/lib/class/SDC
172 dir path=usr/lib/class/TS
173 dir path=usr/lib/crypto
174 dir path=usr/lib/devfsadm group=sys
175 dir path=usr/lib/devfsadm/linkmod group=sys
176 dir path=usr/lib/fs group=sys
177 dir path=usr/lib/fs/autofs group=sys
178 dir path=usr/lib/fs/autofs/$(ARCH64) group=sys
179 dir path=usr/lib/fs/cachefs group=sys
180 dir path=usr/lib/fs/ctfs group=sys
181 dir path=usr/lib/fs/dev group=sys
182 dir path=usr/lib/fs/fd group=sys
183 dir path=usr/lib/fs/hsfs group=sys
184 dir path=usr/lib/fs/lofs group=sys
185 dir path=usr/lib/fs/mntfs group=sys
186 dir path=usr/lib/fs/nfs group=sys
187 dir path=usr/lib/fs/nfs/$(ARCH64) group=sys
188 dir path=usr/lib/fs/objfs group=sys
189 dir path=usr/lib/fs/proc group=sys
190 dir path=usr/lib/fs/sharefs group=sys
191 dir path=usr/lib/fs/tmpfs group=sys
192 dir path=usr/lib/fs/ufs group=sys
```



```
193 dir path=usr/lib/help
194 dir path=usr/lib/help/auths
195 dir path=usr/lib/help/auths/locale
196 dir path=usr/lib/help/auths/locale/C
197 dir path=usr/lib/help/profiles
198 dir path=usr/lib/help/profiles/locale
199 dir path=usr/lib/help/profiles/locale/C
200 dir path=usr/lib/iconv
201 dir path=usr/lib/inet
202 dir path=usr/lib/inet/$(ARCH32)
203 dir path=usr/lib/inet/$(ARCH64)
204 dir path=usr/lib/inet/dhcp
205 dir path=usr/lib/inet/dhcp/nsu
206 dir path=usr/lib/inet/dhcp/svc
207 dir path=usr/lib/locale
208 dir path=usr/lib/locale/C
209 dir path=usr/lib/locale/C/LC_COLLATE
210 dir path=usr/lib/locale/C/LC_CTYPE
211 dir path=usr/lib/locale/C/LC_MESSAGES
212 dir path=usr/lib/locale/C/LC_MONETARY
213 dir path=usr/lib/locale/C/LC_NUMERIC
214 dir path=usr/lib/locale/C/LC_TIME
215 dir path=usr/lib/netsvc group=sys
216 dir path=usr/lib/pci
217 dir path=usr/lib/rcm
218 dir path=usr/lib/rcm/modules
219 dir path=usr/lib/rcm/scripts
220 dir path=usr/lib/reparse
221 dir path=usr/lib/saf
222 dir path=usr/lib/secure
223 dir path=usr/lib/secure/$(ARCH64)
224 dir path=usr/lib/security
225 dir path=usr/lib/sysevent
226 dir path=usr/lib/sysevent/modules
227 dir path=usr/net group=sys
228 dir path=usr/net/nls group=sys
229 dir path=usr/net/servers group=sys
230 dir path=usr/old
231 dir path=usr/platform group=sys
232 dir path=usr/sadm
233 dir path=usr/sadm/bin
234 dir path=usr/sadm/install
235 dir path=usr/sadm/install/scripts
236 dir path=usr/sbin
237 $(i386_ONLY)dir path=usr/sbin/$(ARCH32)
238 dir path=usr/sbin/$(ARCH64)
239 dir path=usr/share
240 dir path=usr/share/doc group=other
241 dir path=usr/share/doc/ksh
242 dir path=usr/share/doc/ksh/images
243 dir path=usr/share/doc/ksh/images/callouts
244 dir path=usr/share/lib
245 dir path=usr/share/lib/mailx
246 dir path=usr/share/lib/pub
247 dir path=usr/share/lib/tabset
248 dir path=usr/share/lib/xml group=sys
249 dir path=usr/share/lib/xml/dtd group=sys
250 dir path=usr/share/lib/xml/style group=sys
251 dir path=usr/share/man
252 dir path=usr/share/man/man1
253 dir path=usr/share/man/man1m
254 dir path=usr/share/man/man4
255 dir path=usr/share/man/man5
256 dir path=usr/share/man/man7d
257 dir path=usr/share/man/man7fs
258 dir path=usr/share/src group=sys
```

```
dir path=var group=sys
dir path=var/adm group=sys mode=0775
dir path=var/adm/exacct group=adm owner=adm
dir path=var/adm/log group=adm owner=adm
dir path=var/adm/streams group=sys
dir path=var/audit group=sys
dir path=var/cores group=sys
dir path=var/cron group=sys
dir path=var/games
dir path=var/idmap group=daemon owner=daemon
dir path=var/inet group=sys
dir path=var/ld
dir path=var/ld/$(ARCH64)
dir path=var/log group=sys
dir path=var/logadm
dir path=var/mail group=mail mode=1777
dir path=var/mail/:saved group=mail mode=0775
dir path=var/news
dir path=var/opt group=sys
dir path=var/preserve mode=1777
dir path=var/run group=sys
dir path=var/sadm group=sys
dir path=var/sadm/system group=sys
dir path=var/sadm/system/admin group=sys
dir path=var/saf
dir path=var/saf/zsmon group=sys
dir path=var/spool
dir path=var/spool/cron group=sys
dir path=var/spool/cron/atjobs group=sys
dir path=var/spool/cron/crontabs group=sys
dir path=var/spool/locks group=uucp owner=uucp
dir path=var/svc group=sys
dir path=var/svc/log group=sys
dir path=var/svc/manifest group=sys
dir path=var/svc/manifest/application group=sys
dir path=var/svc/manifest/application/management group=sys
dir path=var/svc/manifest/application/print group=sys
dir path=var/svc/manifest/application/security group=sys
dir path=var/svc/manifest/device group=sys
dir path=var/svc/manifest/milestone group=sys
dir path=var/svc/manifest/network group=sys
dir path=var/svc/manifest/network/dns group=sys
dir path=var/svc/manifest/network/ipsec group=sys
dir path=var/svc/manifest/network/ldap group=sys
dir path=var/svc/manifest/network/nfs group=sys
dir path=var/svc/manifest/network/nis group=sys
dir path=var/svc/manifest/network/routing group=sys
dir path=var/svc/manifest/network/rpc group=sys
dir path=var/svc/manifest/network/security group=sys
dir path=var/svc/manifest/network/shares group=sys
dir path=var/svc/manifest/network/ssl group=sys
dir path=var/svc/manifest/platform group=sys
$(sparc_ONLY)dir path=var/svc/manifest/platform/sun4u group=sys
$(sparc_ONLY)dir path=var/svc/manifest/platform/sun4v group=sys
dir path=var/svc/manifest/site group=sys
dir path=var/svc/manifest/system group=sys
dir path=var/svc/manifest/system/device group=sys
dir path=var/svc/manifest/system/filesystem group=sys
dir path=var/svc/manifest/system/security group=sys
dir path=var/svc/manifest/system/svc group=sys
dir path=var/svc/profile group=sys
dir path=var/tmp group=sys mode=1777
driver name=dump perms="dump 0660 root sys"
driver name=fssnap \
    policy="ctl read_priv_set=sys_config write_priv_set=sys_config" \
    perms="* 0640 root sys" perms="ctl 0666 root sys"
```



```
driver name=kstat perms="* 0666 root sys"
driver name=ksyms perms="* 0666 root sys"
driver name=logindmux
driver name=ptm clone_perms="ptmx 0666 root sys"
driver name=pts perms="* 0644 root sys" perms="0 0620 root tty" \
    perms="1 0620 root tty" perms="2 0620 root tty" perms="3 0620 root tty"
file path=etc/.login group=sys preserve=renamenew
file path=etc/certs/SUNWObjectCA group=sys
file path=etc/certs/SUNWSolarisCA group=sys
file path=etc/certs/SUNW_SunOS_5.10 group=sys
file path=etc/cron.d/.proto group=sys mode=0744
file path=etc/cron.d/at.deny group=sys preserve=true
file path=etc/cron.d/cron.deny group=sys preserve=true
file path=etc/cron.d/queuedefs group=sys
file path=etc/crypto/certs/CA group=sys
file path=etc/crypto/certs/SUNW_SunOS_5.10 group=sys
file path=etc/crypto/kmf.conf group=sys preserve=true
file path=etc/crypto/pkcs11.conf group=sys preserve=true
file path=etc/datemsk group=sys mode=0444
file path=etc/default/cron group=sys preserve=true
file path=etc/default/devfsadm group=sys preserve=true
file path=etc/default/fs group=sys preserve=true
file path=etc/default/init group=sys preserve=true
file path=etc/default/keyserv group=sys preserve=true
file path=etc/default/login group=sys preserve=true
file path=etc/default/nss group=sys preserve=true
file path=etc/default/passwd group=sys preserve=true
file path=etc/default/su group=sys preserve=true
file path=etc/default/syslogd group=sys preserve=true
file path=etc/default/tar group=sys preserve=true
file path=etc/default/utmpd group=sys preserve=true
file path=etc/dev/reserved_devnames group=sys preserve=true
file path=etc/device.tab group=root mode=0444 preserve=true
file path=etc/dfs/dfstab group=sys preserve=true
file path=etc/dfs/fstypes group=root preserve=true
file path=etc/dfs/sharetab group=root mode=0444 preserve=true
file path=etc/dgroup.tab group=sys mode=0444 preserve=true
file path=etc/dhcp/inittab group=sys preserve=true
file path=etc/dhcp/inittab6 group=sys preserve=true
file path=etc/dumpdates group=sys mode=0664 preserve=true
file path=etc/format.dat group=sys preserve=true
file path=etc/fs/dev/mount mode=0555
file path=etc/fs/hsfs/mount mode=0555
file path=etc/fs/ufs/mount mode=0555
file path=etc/ftpd/ftpusers group=sys preserve=true
file path=etc/group group=sys preserve=true
file path=etc/inet/hosts group=sys preserve=true
file path=etc/inet/inetd.conf group=sys preserve=true
file path=etc/inet/ipaddrsel.conf group=sys preserve=true
file path=etc/inet/netmasks group=sys preserve=true
file path=etc/inet/networks group=sys preserve=true
file path=etc/inet/protocols group=sys preserve=true
file path=etc/inet/services group=sys preserve=true
file path=etc/inet/wanboot.conf.sample group=sys mode=0444
file path=etc/init.d/PRESERVE group=sys mode=0744 preserve=true
file path=etc/init.d/README group=sys preserve=true
file path=etc/init.d/cachefs.daemon group=sys mode=0744 preserve=true
file path=etc/init.d/ldap.client group=sys mode=0744
file path=etc/init.d/nscd group=sys mode=0744
file path=etc/init.d/sysetup group=sys mode=0744 preserve=true
file path=etc/init.d/ufs_quota group=sys mode=0744 preserve=true
file path=etc/inittab group=sys preserve=true
file path=etc/ioctl.syscon group=sys preserve=true
file path=etc/ksh.kshrc group=sys preserve=renameold
file path=etc/logadm.conf group=sys preserve=true timestamp=19700101T000000Z
file path=etc/logindevperm group=sys preserve=true
```

```
391 file path=etc/magic mode=0444
392 file path=etc/mail/mailx.rc preserve=true
393 file path=etc/mailcap preserve=true
394 file path=etc/mime.types preserve=true
395 file path=etc/mnttab group=root mode=0444 preserve=true
396 file path=etc/motd group=sys preserve=true
397 file path=etc/net/ticlts/hosts group=sys
398 file path=etc/net/ticlts/services group=sys preserve=true
399 file path=etc/net/ticots/hosts group=sys
400 file path=etc/net/ticots/services group=sys preserve=true
401 file path=etc/net/ticotsord/hosts group=sys
402 file path=etc/net/ticotsord/services group=sys preserve=true
403 file path=etc/netconfig group=sys preserve=true
404 file path=etc/nscd.conf group=sys preserve=true
405 file path=etc/nsswitch.ad group=sys
406 file path=etc/nsswitch.conf group=sys preserve=true
407 file path=etc/nsswitch.dns group=sys
408 file path=etc/nsswitch.files group=sys
409 file path=etc/nsswitch.ldap group=sys
410 file path=etc/pam.conf group=sys preserve=true
411 file path=etc/passwd group=sys preserve=true
412 file path=etc/profile group=sys preserve=true
413 file path=etc/project group=sys preserve=true
414 file path=etc/rc2.d/README group=sys
415 file path=etc/rc3.d/README group=sys
416 file path=etc/rcS.d/README group=sys
417 file path=etc/remote preserve=true
418 file path=etc/rpc group=sys preserve=true
419 file path=etc/saf/_sactab group=sys preserve=true
420 file path=etc/saf/_sysconfig group=sys preserve=true
421 file path=etc/saf/zsmon/_pmtab group=sys preserve=true
422 file path=etc/security/audit_class group=sys preserve=renamenew
423 file path=etc/security/audit_event group=sys preserve=renamenew
424 file path=etc/security/audit_warn group=sys mode=0740 preserve=renamenew
425 file path=etc/security/auth_attr group=sys preserve=true \
426     timestamp=19700101T000000Z
427 file path=etc/security/auth_attr.d/SUNWcs group=sys
428 file path=etc/security/crypt.conf group=sys preserve=renamenew
429 file path=etc/security/dev/audio mode=0400
430 file path=etc/security/dev/fd0 mode=0400
431 file path=etc/security/dev/sr0 mode=0400
432 file path=etc/security/dev/st0 mode=0400
433 file path=etc/security/dev/st1 mode=0400
434 file path=etc/security/exec_attr group=sys preserve=true \
435     timestamp=19700101T000000Z
436 file path=etc/security/exec_attr.d/SUNWcs group=sys
437 file path=etc/security/kmfpolicy.xml
438 file path=etc/security/lib/audio_clean group=sys mode=0555
439 file path=etc/security/lib/fd_clean group=sys mode=0555
440 file path=etc/security/lib/sr_clean group=sys mode=0555
441 file path=etc/security/lib/st_clean group=sys mode=0555
442 file path=etc/security/policy.conf group=sys preserve=true
443 file path=etc/security/priv_names group=sys preserve=renameold
444 file path=etc/security/prof_attr group=sys preserve=true \
445     timestamp=19700101T000000Z
446 file path=etc/security/prof_attr.d/SUNWcs group=sys
447 file path=etc/shadow group=sys mode=0400 preserve=true
448 file path=etc/skel/.profile group=other preserve=true
449 file path=etc/skel/local.cshrc group=sys preserve=true
450 file path=etc/skel/local.login group=sys preserve=true
451 file path=etc/skel/local.profile group=sys preserve=true
452 file path=etc/svc/profile/generic_limited_net.xml group=sys mode=0444
453 file path=etc/svc/profile/generic_open.xml group=sys mode=0444
454 file path=etc/svc/profile/inetd_generic.xml group=sys mode=0444
455 file path=etc/svc/profile/inetd_upgrade.xml group=sys mode=0444
456 file path=etc/svc/profile/ns_dns.xml group=sys mode=0444
```



```
457 file path=etc/svc/profile/ns_files.xml group=sys mode=0444
458 file path=etc/svc/profile/ns_ldap.xml group=sys mode=0444
459 file path=etc/svc/profile/ns_nis.xml group=sys mode=0444
460 file path=etc/svc/profile/ns_none.xml group=sys mode=0444
461 $(sparc_ONLY)file path=etc/svc/profile/platform_SUNW,SPARC-Enterprise.xml \
462     group=sys mode=0444
463 $(sparc_ONLY)file path=etc/svc/profile/platform_SUNW,Sun-Fire-15000.xml \
464     group=sys mode=0444
465 $(sparc_ONLY)file path=etc/svc/profile/platform_SUNW,Sun-Fire-880.xml \
466     group=sys mode=0444
467 $(sparc_ONLY)file path=etc/svc/profile/platform_SUNW,Sun-Fire.xml group=sys \
468     mode=0444
469 $(sparc_ONLY)file \
470     path=etc/svc/profile/platform_SUNW,Ultra-Enterprise-10000.xml group=sys \
471     mode=0444
472 $(sparc_ONLY)file \
473     path=etc/svc/profile/platform_SUNW,UltraSPARC-IIi-Netract.xml group=sys \
474     mode=0444
475 file path=etc/svc/profile/platform_none.xml group=sys mode=0444
476 $(sparc_ONLY)file path=etc/svc/profile/platform_sun4v.xml group=sys mode=0444
477 file path=etc/sysevent/config/README group=sys mode=0444
478 file path=etc/sysevent/config/SUNW,EC_dr,ESC_dr_req,sysevent.conf group=sys
479 file path=etc/syslog.conf group=sys preserve=true
480 file path=etc/ttydefs group=sys preserve=true
481 file path=etc/ttysrch group=sys preserve=true
482 file path=etc/user_attr group=sys preserve=true timestamp=19700101T000000Z
483 file path=etc/user_attr.d/SUNWcs group=sys
484 file path=etc/vfstab group=sys preserve=true
485 file path=lib/inet/in.mpathd mode=0555
486 file path=lib/inet/ipmgmtd mode=0555
487 file path=lib/inet/netcfgd mode=0555
488 file path=lib/inet/nwamd mode=0555
489 file path=lib/svc/bin/lsvcrun group=sys mode=0555
490 file path=lib/svc/bin/mfstscan group=sys mode=0555
491 file path=lib/svc/bin/restore_repository group=sys mode=0555
492 file path=lib/svc/bin/sqlite group=sys mode=0555
493 file path=lib/svc/bin/svc.configd group=sys mode=0555
494 file path=lib/svc/bin/svc.ipfd group=sys mode=0555
495 file path=lib/svc/bin/svc.startd group=sys mode=0555
496 file path=lib/svc/manifest/milestone/multi-user-server.xml group=sys mode=0444
497 file path=lib/svc/manifest/milestone/multi-user.xml group=sys mode=0444
498 file path=lib/svc/manifest/milestone/name-services.xml group=sys mode=0444
499 file path=lib/svc/manifest/milestone/network.xml group=sys mode=0444
500 file path=lib/svc/manifest/milestone/single-user.xml group=sys mode=0444
501 file path=lib/svc/manifest/milestone/sysconfig.xml group=sys mode=0444
502 file path=lib/svc/manifest/network/dlmgmt.xml group=sys mode=0444
503 file path=lib/svc/manifest/network/dns/client.xml group=sys mode=0444
504 file path=lib/svc/manifest/network/dns/install.xml group=sys mode=0444
505 file path=lib/svc/manifest/network/forwarding.xml group=sys mode=0444
506 file path=lib/svc/manifest/network/inetd-upgrade.xml group=sys mode=0444
507 file path=lib/svc/manifest/network/inetd.xml group=sys mode=0444
508 file path=lib/svc/manifest/network/ipsec/ike.xml group=sys mode=0444
509 file path=lib/svc/manifest/network/ipsec/ipsecalgs.xml group=sys mode=0444
510 file path=lib/svc/manifest/network/ipsec/manual-key.xml group=sys mode=0444
511 file path=lib/svc/manifest/network/ipsec/policy.xml group=sys mode=0444
512 file path=lib/svc/manifest/network/ldap/client.xml group=sys mode=0444
513 file path=lib/svc/manifest/network/network-initial.xml group=sys mode=0444
514 file path=lib/svc/manifest/network/network-install.xml group=sys mode=0444
515 file path=lib/svc/manifest/network/network-ipmgmt.xml group=sys mode=0444
516 file path=lib/svc/manifest/network/network-ipqos.xml group=sys mode=0444
517 file path=lib/svc/manifest/network/network-iptun.xml group=sys mode=0444
518 file path=lib/svc/manifest/network/network-location.xml group=sys mode=0444
519 file path=lib/svc/manifest/network/network-loopback.xml group=sys mode=0444
520 file path=lib/svc/manifest/network/network-netcfg.xml group=sys mode=0444
521 file path=lib/svc/manifest/network/network-netmask.xml group=sys mode=0444
522 file path=lib/svc/manifest/network/network-physical.xml group=sys mode=0444
```

```
523 file path=lib/svc/manifest/network/network-routing-setup.xml group=sys \
524     mode=0444
525 file path=lib/svc/manifest/network/network-service.xml group=sys mode=0444
526 file path=lib/svc/manifest/network/routing/legacy-routing.xml group=sys \
527     mode=0444
528 file path=lib/svc/manifest/network/rpc/bind.xml group=sys mode=0444
529 file path=lib/svc/manifest/network/rpc/keyserv.xml group=sys mode=0444
530 file path=lib/svc/manifest/network/shares/group.xml group=sys mode=0444
531 file path=lib/svc/manifest/network/shares/reparsed.xml group=sys mode=0444
532 file path=lib/svc/manifest/network/socket-filter-kssl.xml group=sys mode=0444
533 file path=lib/svc/manifest/network/ssl/kssl-proxy.xml group=sys mode=0444
534 file path=lib/svc/manifest/system/auditd.xml group=sys mode=0444
535 file path=lib/svc/manifest/system/auditset.xml group=sys mode=0444
536 file path=lib/svc/manifest/system/boot-archive-update.xml group=sys mode=0444
537 file path=lib/svc/manifest/system/boot-archive.xml group=sys mode=0444
538 file path=lib/svc/manifest/system/boot-config.xml group=sys mode=0444
539 file path=lib/svc/manifest/system/consadm.xml group=sys mode=0444
540 file path=lib/svc/manifest/system/console-login.xml group=sys mode=0444
541 file path=lib/svc/manifest/system/coreadm.xml group=sys mode=0444
542 file path=lib/svc/manifest/system/cron.xml group=sys mode=0444
543 file path=lib/svc/manifest/system/cryptosvc.xml group=sys mode=0444
544 file path=lib/svc/manifest/system/device/allocate.xml group=sys mode=0444
545 file path=lib/svc/manifest/system/device/devices-audio.xml group=sys mode=0444
546 file path=lib/svc/manifest/system/device/devices-local.xml group=sys mode=0444
547 file path=lib/svc/manifest/system/device/mpxio-upgrade.xml group=sys mode=0444
548 file path=lib/svc/manifest/system/early-manifest-import.xml group=sys \
549     mode=0444
550 file path=lib/svc/manifest/system/extended-accounting.xml group=sys mode=0444
551 file path=lib/svc/manifest/system/filesystem/local-fs.xml group=sys mode=0444
552 file path=lib/svc/manifest/system/filesystem/minimal-fs.xml group=sys \
553     mode=0444
554 file path=lib/svc/manifest/system/filesystem/root-fs.xml group=sys mode=0444
555 file path=lib/svc/manifest/system/filesystem/usr-fs.xml group=sys mode=0444
556 $(i386_ONLY)file path=lib/svc/manifest/system/hostid.xml group=sys mode=0444
557 file path=lib/svc/manifest/system/hotplug.xml group=sys mode=0444
558 file path=lib/svc/manifest/system/identity.xml group=sys mode=0444
559 file path=lib/svc/manifest/system/idmap.xml group=sys mode=0444
560 file path=lib/svc/manifest/system/keymap.xml group=sys mode=0444
561 file path=lib/svc/manifest/system/logadm-upgrade.xml group=sys mode=0444
562 file path=lib/svc/manifest/system/manifest-import.xml group=sys mode=0444
563 file path=lib/svc/manifest/system/name-service-cache.xml group=sys mode=0444
564 file path=lib/svc/manifest/system/pfexecd.xml group=sys mode=0444
565 file path=lib/svc/manifest/system/rbac.xml group=sys mode=0444
566 file path=lib/svc/manifest/system/rmtmpfiles.xml group=sys mode=0444
567 file path=lib/svc/manifest/system/sac.xml group=sys mode=0444
568 file path=lib/svc/manifest/system/svc/global.xml group=sys mode=0444
569 file path=lib/svc/manifest/system/svc/restarter.xml group=sys mode=0444
570 file path=lib/svc/manifest/system/system-log.xml group=sys mode=0444
571 file path=lib/svc/manifest/system/utmp.xml group=sys mode=0444
572 file path=lib/svc/manifest/system/vtdaemon.xml group=sys mode=0444
573 file path=lib/svc/method/boot-archive mode=0555
574 file path=lib/svc/method/boot-archive-update mode=0555
575 file path=lib/svc/method/console-login mode=0555
576 file path=lib/svc/method/devices-audio mode=0555
577 file path=lib/svc/method/devices-local mode=0555
578 file path=lib/svc/method/dns-install mode=0555
579 file path=lib/svc/method/fs-local mode=0555
580 file path=lib/svc/method/fs-minimal mode=0555
581 file path=lib/svc/method/fs-root mode=0555
582 file path=lib/svc/method/fs-usr mode=0555
583 file path=lib/svc/method/identity-domain mode=0555
584 file path=lib/svc/method/identity-node mode=0555
585 file path=lib/svc/method/inetd-upgrade mode=0555
586 file path=lib/svc/method/keymap mode=0555
587 file path=lib/svc/method/ldap-client mode=0555
588 file path=lib/svc/method/logadm-upgrade mode=0555
```



```
589 file path=lib/svc/method/manifest-import mode=0555
590 file path=lib/svc/method/mpxio-upgrade mode=0555
591 file path=lib/svc/method/net-init mode=0555
592 file path=lib/svc/method/net-install mode=0555
593 file path=lib/svc/method/net-ipmgmt mode=0555
594 file path=lib/svc/method/net-ipqos mode=0555
595 file path=lib/svc/method/net-iptun mode=0555
596 file path=lib/svc/method/net-loc mode=0555
597 file path=lib/svc/method/net-loopback mode=0555
598 file path=lib/svc/method/net-netmask mode=0555
599 file path=lib/svc/method/net-nwam mode=0555
600 file path=lib/svc/method/net-physical mode=0555
601 file path=lib/svc/method/net-routing-setup mode=0555
602 file path=lib/svc/method/net-svc mode=0555
603 file path=lib/svc/method/rmtmpfiles mode=0555
604 file path=lib/svc/method/rpc-bind mode=0555
605 file path=lib/svc/method/svc-allocate mode=0555
606 file path=lib/svc/method/svc-auditd mode=0555
607 file path=lib/svc/method/svc-auditset mode=0555
608 file path=lib/svc/method/svc-boot-config mode=0555
609 file path=lib/svc/method/svc-consadm mode=0555
610 file path=lib/svc/method/svc-cron mode=0555
611 file path=lib/svc/method/svc-dlmgmtd mode=0555
612 file path=lib/svc/method/svc-forwarding mode=0555
613 $(i386_ONLY)file path=lib/svc/method/svc-hostid mode=0555
614 file path=lib/svc/method/svc-hotplug mode=0555
615 file path=lib/svc/method/svc-legacy-routing mode=0555
616 file path=lib/svc/method/svc-nscd mode=0555
617 file path=lib/svc/method/svc-rbac mode=0555
618 file path=lib/svc/method/svc-sockfilter mode=0555
619 file path=lib/svc/method/svc-utmpd mode=0555
620 file path=lib/svc/method/system-log mode=0555
621 file path=lib/svc/method/vtdaemon mode=0555
622 file path=lib/svc/method/yp mode=0555
623 # global.db is not needed in non-global zones, and it's pretty large.
624 file path=lib/svc/seed/global.db group=sys mode=0444 \
625     variant.opensolaris.zone=global
626 # symmetrically, nonglobal.db is not needed in global zones.
627 file path=lib/svc/seed/nonglobal.db group=sys mode=0444 \
628     variant.opensolaris.zone=nonglobal
629 file path=lib/svc/share/README mode=0444
630 file path=lib/svc/share/fs_include.sh mode=0444
631 file path=lib/svc/share/ipf_include.sh mode=0444
632 file path=lib/svc/share/mfsthistory mode=0444
633 file path=lib/svc/share/net_include.sh mode=0444
634 file path=lib/svc/share/routing_include.sh mode=0444
635 file path=lib/svc/share/smf_include.sh mode=0444
636 file path=root/.bashrc group=root preserve=true
637 file path=root/.profile group=root preserve=true
638 file path=sbin/autopush mode=0555
639 $(i386_ONLY)file path=sbin/biosdev mode=0555
640 file path=sbin/bootadm mode=0555
641 file path=sbin/cryptoadm mode=0555
642 file path=sbin/devprop mode=0555
643 file path=sbin/dhcpagent mode=0555
644 file path=sbin/dhcpinfo mode=0555
645 file path=sbin/dlmgmtd mode=0555
646 file path=sbin/fdisk mode=0555
647 file path=sbin/fiocompress mode=0555
648 file path=sbin/hostconfig mode=0555
649 file path=sbin/ifconfig mode=0555
650 file path=sbin/ifparse mode=0555
651 file path=sbin/init group=sys mode=0555
652 $(i386_ONLY)file path=sbin/installgrub group=sys mode=0555
653 file path=sbin/ipmpstat mode=0555
654 file path=sbin/mount mode=0555
```

```
655 file path=sbin/mountall group=sys mode=0555
656 file path=sbin/netstrategy mode=0555
657 file path=sbin/rc0 group=sys mode=0744
658 file path=sbin/rc1 group=sys mode=0744
659 file path=sbin/rc2 group=sys mode=0744
660 file path=sbin/rc3 group=sys mode=0744
661 file path=sbin/rcS group=sys mode=0744
662 file path=sbin/route mode=0555
663 file path=sbin/routeadm mode=0555
664 file path=sbin/soconfig mode=0555
665 file path=sbin/su.static group=sys mode=0555
666 file path=sbin/sulogin mode=0555
667 file path=sbin/swapadd group=sys mode=0744
668 file path=sbin/sync mode=0555
669 file path=sbin/tzreload mode=0555
670 file path=sbin/uadmin group=sys mode=0555
671 file path=sbin/umount mode=0555
672 file path=sbin/umountall group=sys mode=0555
673 file path=sbin/uname mode=0555
674 file path=sbin/wusbadm mode=0555
675 file path=sbin/zonename mode=0555
676 $(i386_ONLY)file path=usr/bin/$(ARCH32)/amt mode=0555
677 file path=usr/bin/$(ARCH32)/decrypt mode=0555
678 file path=usr/bin/$(ARCH32)/digest mode=0555
679 file path=usr/bin/$(ARCH32)/ksh93 mode=0555
680 $(i386_ONLY)file path=usr/bin/$(ARCH32)/newtask group=sys mode=4555
681 $(i386_ONLY)file path=usr/bin/$(ARCH32)/nohup mode=0555
682 $(i386_ONLY)file path=usr/bin/$(ARCH32)/prctl mode=0555
683 $(i386_ONLY)file path=usr/bin/$(ARCH32)/prstat mode=0555
684 $(i386_ONLY)file path=usr/bin/$(ARCH32)/ps mode=0555
685 file path=usr/bin/$(ARCH32)/savecore mode=0555
686 $(i386_ONLY)file path=usr/bin/$(ARCH32)/setuname mode=0555
687 $(i386_ONLY)file path=usr/bin/$(ARCH32)/uptime mode=4555
688 file path=usr/bin/$(ARCH64)/amt mode=0555
689 file path=usr/bin/$(ARCH64)/crle mode=0555
690 file path=usr/bin/$(ARCH64)/decrypt mode=0555
691 file path=usr/bin/$(ARCH64)/digest mode=0555
692 file path=usr/bin/$(ARCH64)/ksh93 mode=0555
693 file path=usr/bin/$(ARCH64)/ls mode=0555
694 file path=usr/bin/$(ARCH64)/moe mode=0555
695 file path=usr/bin/$(ARCH64)/newtask group=sys mode=4555
696 file path=usr/bin/$(ARCH64)/nohup mode=0555
697 file path=usr/bin/$(ARCH64)/prctl mode=0555
698 file path=usr/bin/$(ARCH64)/prstat mode=0555
699 file path=usr/bin/$(ARCH64)/ps mode=0555
700 file path=usr/bin/$(ARCH64)/savecore mode=0555
701 file path=usr/bin/$(ARCH64)/setuname mode=0555
702 file path=usr/bin/$(ARCH64)/uptime mode=4555
703 $(i386_ONLY)file path=usr/bin/addbadsec mode=0555
704 file path=usr/bin/alias mode=0555
705 file path=usr/bin/amt mode=0555
706 file path=usr/bin/arch mode=0555
707 file path=usr/bin/at group=sys mode=4755
708 file path=usr/bin/atq group=sys mode=4755
709 file path=usr/bin/atrm group=sys mode=4755
710 file path=usr/bin/auths mode=0555
711 file path=usr/bin/basename mode=0555
712 file path=usr/bin/busstat mode=0555
713 file path=usr/bin/captoinfo mode=0555
714 file path=usr/bin/cat mode=0555
715 file path=usr/bin/chgrp mode=0555
716 file path=usr/bin/chmod mode=0555
717 file path=usr/bin/chown mode=0555
718 file path=usr/bin/ckdate mode=0555
719 file path=usr/bin/ckgid mode=0555
720 file path=usr/bin/ckint mode=0555
```



```
721 file path=usr/bin/ckitem mode=0555
722 file path=usr/bin/ckkeywd mode=0555
723 file path=usr/bin/ckpath mode=0555
724 file path=usr/bin/ckrange mode=0555
725 file path=usr/bin/ckstr mode=0555
726 file path=usr/bin/cktime mode=0555
727 file path=usr/bin/ckuid mode=0555
728 file path=usr/bin/ckyorn mode=0555
729 file path=usr/bin/clear mode=0555
730 file path=usr/bin/coreadm mode=0555
731 file path=usr/bin/cp mode=0555
732 file path=usr/bin/cpio mode=0555
733 file path=usr/bin/crle mode=0555
734 file path=usr/bin/crontab mode=4555
735 file path=usr/bin/crypt mode=0555
736 file path=usr/bin/csh mode=0555
737 file path=usr/bin/ctrun mode=0555
738 file path=usr/bin/ctstat mode=0555
739 file path=usr/bin/ctwatch mode=0555
740 file path=usr/bin/date mode=0555
741 file path=usr/bin/dd mode=0555
742 file path=usr/bin/devattr mode=0555
743 file path=usr/bin/devfree mode=0555
744 file path=usr/bin/devreserv mode=0555
745 file path=usr/bin/dirname mode=0555
746 $(i386_ONLY)file path=usr/bin/diskscan mode=0555
747 file path=usr/bin/domainname mode=0555
748 file path=usr/bin/du mode=0555
749 file path=usr/bin/dumpcs mode=0555
750 file path=usr/bin/dumpkeys mode=0555
751 file path=usr/bin/echo mode=0555
752 file path=usr/bin/ed mode=0555
753 file path=usr/bin/egrep mode=0555
754 file path=usr/bin/eject mode=0555
755 file path=usr/bin/env mode=0555
756 file path=usr/bin/expr mode=0555
757 file path=usr/bin/false mode=0555
758 file path=usr/bin/fdetach mode=0555
759 file path=usr/bin/fdformat mode=4555
760 file path=usr/bin/fgrep mode=0555
761 file path=usr/bin/file mode=0555
762 file path=usr/bin/find mode=0555
763 file path=usr/bin/fmt mode=0555
764 file path=usr/bin/fmtmsg mode=0555
765 file path=usr/bin/fold mode=0555
766 file path=usr/bin/fsstat mode=0555
767 file path=usr/bin/geniconvtbl mode=0555
768 file path=usr/bin/getconf mode=0555
769 file path=usr/bin/getdev mode=0555
770 file path=usr/bin/getdgrp mode=0555
771 file path=usr/bin/getent mode=0555
772 file path=usr/bin/getfacl mode=0555
773 file path=usr/bin/getopt mode=0555
774 file path=usr/bin/gettext mode=0555
775 file path=usr/bin/getvol mode=0555
776 file path=usr/bin/grep mode=0555
777 file path=usr/bin/groups mode=0555
778 file path=usr/bin/head mode=0555
779 file path=usr/bin/hostid mode=0555
780 file path=usr/bin/hostname mode=0555
781 file path=usr/bin/i286 mode=0555
782 file path=usr/bin/iconv mode=0555
783 file path=usr/bin/id mode=0555
784 file path=usr/bin/infocmp mode=0555
785 file path=usr/bin/iostat mode=0555
786 file path=usr/bin/isainfo mode=0555
```

```
 787 file path=usr/bin/isalist mode=0555
 788 file path=usr/bin/itu mode=0555
 789 file path=usr/bin/kbd mode=0555
 790 file path=usr/bin/keylogin mode=0555
 791 file path=usr/bin/keylogout mode=0555
 792 file path=usr/bin/kmfcfg mode=0555
 793 file path=usr/bin/kvmstat mode=0555
 794 file path=usr/bin/line mode=0555
 795 file path=usr/bin/listdgrp mode=0555
 796 file path=usr/bin/listusers mode=0555
 797 file path=usr/bin/loadkeys mode=0555
 798 file path=usr/bin/logger mode=0555
 799 file path=usr/bin/login mode=4555
 800 file path=usr/bin/logins mode=0750
 801 file path=usr/bin/ls mode=0555
 802 file path=usr/bin/m4 mode=0555
 803 file path=usr/bin/mach mode=0555
 804 file path=usr/bin/mail group=mail mode=2511
 805 file path=usr/bin/mailx group=mail mode=2511
 806 file path=usr/bin/makedev mode=0555
 807 file path=usr/bin/mesg mode=0555
 808 file path=usr/bin/mkbootmedia mode=0555
 809 file path=usr/bin/mkdir mode=0555
 810 file path=usr/bin/mkpwdict mode=0555
 811 file path=usr/bin/mktemp mode=0555
 812 file path=usr/bin/moe mode=0555
 813 file path=usr/bin/more mode=0555
 814 file path=usr/bin/mpstat mode=0555
 815 file path=usr/bin/mt mode=0555
 816 file path=usr/bin/netstat mode=0555
 817 file path=usr/bin/newgrp group=sys mode=4755
 818 file path=usr/bin/nice mode=0555
 819 file path=usr/bin/optisa mode=0555
 820 file path=usr/bin/pagesize mode=0555
 821 file path=usr/bin/passwd group=sys mode=6555
 822 file path=usr/bin/pathchk mode=0555
 823 file path=usr/bin/pax mode=0555
 824 file path=usr/bin/pfexec mode=0555
 825 file path=usr/bin/pg mode=0555
 826 file path=usr/bin/pgrep mode=0555
 827 file path=usr/bin/pkg2du mode=0555
 828 file path=usr/bin/pktool mode=0555
 829 file path=usr/bin/pr mode=0555
 830 file path=usr/bin/printf mode=0555
 831 file path=usr/bin/priocntl mode=0555
 832 file path=usr/bin/profiles mode=0555
 833 file path=usr/bin/projects mode=0555
 834 file path=usr/bin/putdev mode=0555
 835 file path=usr/bin/putdgrp mode=0555
 836 file path=usr/bin/pwd mode=0555
 837 file path=usr/bin/renice mode=0555
 838 file path=usr/bin/rm mode=0555
 839 file path=usr/bin/rmdir mode=0555
 840 file path=usr/bin/roles mode=0555
 841 file path=usr/bin/rpcinfo mode=0555
 842 file path=usr/bin/runat mode=0555
 843 file path=usr/bin/script mode=0555
 844 file path=usr/bin/sed mode=0555
 845 file path=usr/bin/setfacl mode=0555
 846 file path=usr/bin/setpgrp group=sys mode=0555
 847 file path=usr/bin/settime mode=0555
 848 file path=usr/bin/shcomp mode=0555
 849 file path=usr/bin/strchg group=root mode=0555
 850 file path=usr/bin/strconf group=root mode=0555
 851 file path=usr/bin/stty mode=0555
 852 file path=usr/bin/su group=sys mode=4555
```



```
 853 file path=usr/bin/svcprop mode=0555
 854 file path=usr/bin/svcs mode=0555
 855 file path=usr/bin/tabs mode=0555
 856 file path=usr/bin/tail mode=0555
 857 file path=usr/bin/tic mode=0555
 858 file path=usr/bin/time mode=0555
 859 file path=usr/bin/tip mode=4511 owner=uucp
 860 file path=usr/bin/tpmadm mode=0555
 861 file path=usr/bin/tput mode=0555
 862 file path=usr/bin/tr mode=0555
 863 file path=usr/bin/true mode=0555
 864 file path=usr/bin/tty mode=0555
 865 file path=usr/bin/tzselect mode=0555
 866 file path=usr/bin/updatemedia mode=0555
 867 file path=usr/bin/userattr mode=0555
 868 file path=usr/bin/vmstat mode=0555
 869 file path=usr/bin/which mode=0555
 870 file path=usr/bin/who mode=0555
 871 file path=usr/bin/wracct mode=0555
 872 file path=usr/bin/write group=tty mode=2555
 873 file path=usr/bin/xargs mode=0555
 874 file path=usr/bin/xstr mode=0555
 875 file path=usr/has/bin/edit mode=0555
 876 file path=usr/has/bin/sh mode=0555
 877 file path=usr/has/man/man1has/edit.1has
 878 file path=usr/has/man/man1has/ex.1has
 879 file path=usr/has/man/man1has/sh.1has
 880 file path=usr/has/man/man1has/vi.1has
 881 file path=usr/kernel/drv/$(ARCH64)/dump group=sys
 882 file path=usr/kernel/drv/$(ARCH64)/fssnap group=sys
 883 file path=usr/kernel/drv/$(ARCH64)/kstat group=sys
 884 file path=usr/kernel/drv/$(ARCH64)/ksyms group=sys
 885 file path=usr/kernel/drv/$(ARCH64)/logindmux group=sys
 886 file path=usr/kernel/drv/$(ARCH64)/ptm group=sys
 887 file path=usr/kernel/drv/$(ARCH64)/pts group=sys
 888 $(i386_ONLY)file path=usr/kernel/drv/dump group=sys
 889 file path=usr/kernel/drv/dump.conf group=sys
 890 $(i386_ONLY)file path=usr/kernel/drv/fssnap group=sys
 891 file path=usr/kernel/drv/fssnap.conf group=sys
 892 $(i386_ONLY)file path=usr/kernel/drv/kstat group=sys
 893 file path=usr/kernel/drv/kstat.conf group=sys
 894 $(i386_ONLY)file path=usr/kernel/drv/ksyms group=sys
 895 file path=usr/kernel/drv/ksyms.conf group=sys
 896 $(i386_ONLY)file path=usr/kernel/drv/logindmux group=sys
 897 file path=usr/kernel/drv/logindmux.conf group=sys
 898 $(i386_ONLY)file path=usr/kernel/drv/ptm group=sys
 899 file path=usr/kernel/drv/ptm.conf group=sys
 900 $(i386_ONLY)file path=usr/kernel/drv/pts group=sys
 901 file path=usr/kernel/drv/pts.conf group=sys
 902 file path=usr/kernel/exec/$(ARCH64)/javaexec group=sys mode=0755
 903 file path=usr/kernel/exec/$(ARCH64)/shbinexec group=sys mode=0755
 904 $(i386_ONLY)file path=usr/kernel/exec/javaexec group=sys mode=0755
 905 $(i386_ONLY)file path=usr/kernel/exec/shbinexec group=sys mode=0755
 906 file path=usr/kernel/fs/$(ARCH64)/fdfs group=sys mode=0755
 907 file path=usr/kernel/fs/$(ARCH64)/pcfs group=sys mode=0755
 908 $(i386_ONLY)file path=usr/kernel/fs/fdfs group=sys mode=0755
 909 $(i386_ONLY)file path=usr/kernel/fs/pcfs group=sys mode=0755
 910 file path=usr/kernel/sched/$(ARCH64)/FX group=sys mode=0755
 911 file path=usr/kernel/sched/$(ARCH64)/FX_DPTBL group=sys mode=0755
 912 file path=usr/kernel/sched/$(ARCH64)/IA group=sys mode=0755
 913 file path=usr/kernel/sched/$(ARCH64)/RT group=sys mode=0755
 914 file path=usr/kernel/sched/$(ARCH64)/RT_DPTBL group=sys mode=0755
 915 $(i386_ONLY)file path=usr/kernel/sched/FX group=sys mode=0755
 916 $(i386_ONLY)file path=usr/kernel/sched/FX_DPTBL group=sys mode=0755
 917 $(i386_ONLY)file path=usr/kernel/sched/IA group=sys mode=0755
 918 $(i386_ONLY)file path=usr/kernel/sched/RT group=sys mode=0755
```

```
 919 $(i386_ONLY)file path=usr/kernel/sched/RT_DPTBL group=sys mode=0755
 920 file path=usr/kernel/strmod/$(ARCH64)/cryptmod group=sys mode=0755
 921 file path=usr/kernel/strmod/$(ARCH64)/rlmod group=sys mode=0755
 922 file path=usr/kernel/strmod/$(ARCH64)/telmod group=sys mode=0755
 923 $(i386_ONLY)file path=usr/kernel/strmod/cryptmod group=sys mode=0755
 924 $(i386_ONLY)file path=usr/kernel/strmod/rlmod group=sys mode=0755
 925 $(i386_ONLY)file path=usr/kernel/strmod/telmod group=sys mode=0755
 926 file path=usr/kernel/sys/$(ARCH64)/acctctl group=sys mode=0755
 927 file path=usr/kernel/sys/$(ARCH64)/exacctsys group=sys mode=0755
 928 file path=usr/kernel/sys/$(ARCH64)/sysacct group=sys mode=0755
 929 $(i386_ONLY)file path=usr/kernel/sys/acctctl group=sys mode=0755
 930 $(i386_ONLY)file path=usr/kernel/sys/exacctsys group=sys mode=0755
 931 $(i386_ONLY)file path=usr/kernel/sys/sysacct group=sys mode=0755
 932 file path=usr/kvm/README group=sys
 933 file path=usr/lib/$(ARCH64)/libshare.so.1
 934 file path=usr/lib/audit/audit_record_attr mode=0444
 935 file path=usr/lib/calprog mode=0555
 936 file path=usr/lib/class/FX/FXdispadmin mode=0555
 937 file path=usr/lib/class/FX/FXpriocntl mode=0555
 938 file path=usr/lib/class/IA/IAdispadmin mode=0555
 939 file path=usr/lib/class/IA/IApriocntl mode=0555
 940 file path=usr/lib/class/RT/RTdispadmin mode=0555
 941 file path=usr/lib/class/RT/RTpriocntl mode=0555
 942 file path=usr/lib/class/SDC/SDCdispadmin mode=0555
 943 file path=usr/lib/class/SDC/SDCpriocntl mode=0555
 944 file path=usr/lib/class/TS/TSdispadmin mode=0555
 945 file path=usr/lib/class/TS/TSpriocntl mode=0555
 946 file path=usr/lib/devfsadm/linkmod/SUNW_audio_link.so group=sys
 947 file path=usr/lib/devfsadm/linkmod/SUNW_cfg_link.so group=sys
 948 file path=usr/lib/devfsadm/linkmod/SUNW_disk_link.so group=sys
 949 file path=usr/lib/devfsadm/linkmod/SUNW_fssnap_link.so group=sys
 950 file path=usr/lib/devfsadm/linkmod/SUNW_ieee1394_link.so group=sys
 951 file path=usr/lib/devfsadm/linkmod/SUNW_lofi_link.so group=sys
 952 file path=usr/lib/devfsadm/linkmod/SUNW_md_link.so group=sys
 953 file path=usr/lib/devfsadm/linkmod/SUNW_misc_link.so group=sys
 954 file path=usr/lib/devfsadm/linkmod/SUNW_misc_link_$(ARCH).so group=sys
 955 file path=usr/lib/devfsadm/linkmod/SUNW_port_link.so group=sys
 956 file path=usr/lib/devfsadm/linkmod/SUNW_ramdisk_link.so group=sys
 957 file path=usr/lib/devfsadm/linkmod/SUNW_sgen_link.so group=sys
 958 file path=usr/lib/devfsadm/linkmod/SUNW_smp_link.so group=sys
 959 file path=usr/lib/devfsadm/linkmod/SUNW_tape_link.so group=sys
 960 file path=usr/lib/devfsadm/linkmod/SUNW_usb_link.so group=sys
 961 $(i386_ONLY)file path=usr/lib/devfsadm/linkmod/SUNW_xen_link.so group=sys
 962 file path=usr/lib/diffh mode=0555
 963 file path=usr/lib/expreserve mode=0555
 964 file path=usr/lib/exrecover mode=0555
 965 file path=usr/lib/fs/cachefs/cachefsd mode=0555
 966 file path=usr/lib/fs/cachefs/cachefslog mode=0555
 967 file path=usr/lib/fs/cachefs/cachefspack mode=0555
 968 file path=usr/lib/fs/cachefs/cachefsstat mode=0555
 969 file path=usr/lib/fs/cachefs/cachefswssize mode=0555
 970 file path=usr/lib/fs/cachefs/cfsadmin mode=0555
 971 file path=usr/lib/fs/cachefs/cfsfstype mode=0555
 972 file path=usr/lib/fs/cachefs/cfstagchk mode=0555
 973 file path=usr/lib/fs/cachefs/dfshares mode=0555
 974 file path=usr/lib/fs/cachefs/fsck mode=0555
 975 file path=usr/lib/fs/cachefs/mount mode=0555
 976 file path=usr/lib/fs/cachefs/share mode=0555
 977 file path=usr/lib/fs/cachefs/umount mode=0555
 978 file path=usr/lib/fs/cachefs/unshare mode=0555
 979 file path=usr/lib/fs/ctfs/mount mode=0555
 980 file path=usr/lib/fs/fd/mount mode=0555
 981 file path=usr/lib/fs/hsfs/fstyp.so.1 mode=0555
 982 file path=usr/lib/fs/hsfs/labelit mode=0555
 983 file path=usr/lib/fs/lofs/mount mode=0555
 984 file path=usr/lib/fs/mntfs/mount mode=0555
```



```
 985 file path=usr/lib/fs/objfs/mount mode=0555
 986 file path=usr/lib/fs/proc/mount mode=0555
 987 file path=usr/lib/fs/sharefs/mount mode=0555
 988 file path=usr/lib/fs/tmpfs/mount mode=0555
 989 file path=usr/lib/fs/ufs/clri mode=0555
 990 file path=usr/lib/fs/ufs/df mode=0555
 991 file path=usr/lib/fs/ufs/edquota mode=0555
 992 file path=usr/lib/fs/ufs/ff mode=0555
 993 file path=usr/lib/fs/ufs/fsck mode=0555
 994 file path=usr/lib/fs/ufs/fsckall mode=0555
 995 file path=usr/lib/fs/ufs/fsdb mode=0555
 996 file path=usr/lib/fs/ufs/fsirand mode=0555
 997 file path=usr/lib/fs/ufs/fssnap mode=0555
 998 file path=usr/lib/fs/ufs/fstyp.so.1 mode=0555
 999 file path=usr/lib/fs/ufs/labelit mode=0555
1000 file path=usr/lib/fs/ufs/lockfs mode=0555
1001 file path=usr/lib/fs/ufs/mkfs mode=0555
1002 file path=usr/lib/fs/ufs/ncheck mode=0555
1003 file path=usr/lib/fs/ufs/newfs mode=0555
1004 file path=usr/lib/fs/ufs/quot mode=0555
1005 file path=usr/lib/fs/ufs/quota mode=4555
1006 file path=usr/lib/fs/ufs/quotacheck mode=0555
1007 file path=usr/lib/fs/ufs/quotaoff mode=0555
1008 file path=usr/lib/fs/ufs/repquota mode=0555
1009 file path=usr/lib/fs/ufs/tunefs mode=0555
1010 file path=usr/lib/fs/ufs/ufsdump mode=4555
1011 file path=usr/lib/fs/ufs/ufsrestore mode=4555
1012 file path=usr/lib/fs/ufs/volcopy mode=0555
1013 file path=usr/lib/getoptcvt mode=0555
1014 file path=usr/lib/help/auths/locale/C/AllSolAuthsHeader.html
1015 file path=usr/lib/help/auths/locale/C/AuditHeader.html
1016 file path=usr/lib/help/auths/locale/C/AuthJobsAdmin.html
1017 file path=usr/lib/help/auths/locale/C/AuthJobsUser.html
1018 file path=usr/lib/help/auths/locale/C/AuthProfmgrAssign.html
1019 file path=usr/lib/help/auths/locale/C/AuthProfmgrDelegate.html
1020 file path=usr/lib/help/auths/locale/C/AuthProfmgrExecattrWrite.html
1021 file path=usr/lib/help/auths/locale/C/AuthProfmgrRead.html
1022 file path=usr/lib/help/auths/locale/C/AuthProfmgrWrite.html
1023 file path=usr/lib/help/auths/locale/C/AuthReadNDMP.html
1024 file path=usr/lib/help/auths/locale/C/AuthReadSMB.html
1025 file path=usr/lib/help/auths/locale/C/AuthRoleAssign.html
1026 file path=usr/lib/help/auths/locale/C/AuthRoleDelegate.html
1027 file path=usr/lib/help/auths/locale/C/AuthRoleWrite.html
1028 file path=usr/lib/help/auths/locale/C/BindStates.html
1029 file path=usr/lib/help/auths/locale/C/DevAllocHeader.html
1030 file path=usr/lib/help/auths/locale/C/DevAllocate.html
1031 file path=usr/lib/help/auths/locale/C/DevConfig.html
1032 file path=usr/lib/help/auths/locale/C/DevGrant.html
1033 file path=usr/lib/help/auths/locale/C/DevRevoke.html
1034 file path=usr/lib/help/auths/locale/C/DhcpmgrHeader.html
1035 file path=usr/lib/help/auths/locale/C/DhcpmgrWrite.html
1036 file path=usr/lib/help/auths/locale/C/HotplugHeader.html
1037 file path=usr/lib/help/auths/locale/C/HotplugModify.html
1038 file path=usr/lib/help/auths/locale/C/IdmapRules.html
1039 file path=usr/lib/help/auths/locale/C/JobHeader.html
1040 file path=usr/lib/help/auths/locale/C/JobsGrant.html
1041 file path=usr/lib/help/auths/locale/C/LinkSecurity.html
1042 file path=usr/lib/help/auths/locale/C/LoginEnable.html
1043 file path=usr/lib/help/auths/locale/C/LoginHeader.html
1044 file path=usr/lib/help/auths/locale/C/LoginRemote.html
1045 file path=usr/lib/help/auths/locale/C/NetworkAutoconfRead.html
1046 file path=usr/lib/help/auths/locale/C/NetworkAutoconfSelect.html
1047 file path=usr/lib/help/auths/locale/C/NetworkAutoconfWlan.html
1048 file path=usr/lib/help/auths/locale/C/NetworkAutoconfWrite.html
1049 file path=usr/lib/help/auths/locale/C/NetworkHeader.html
1050 file path=usr/lib/help/auths/locale/C/NetworkILBconf.html
```

```
1051 file path=usr/lib/help/auths/locale/C/NetworkILBenable.html
1052 file path=usr/lib/help/auths/locale/C/NetworkInterfaceConfig.html
1053 file path=usr/lib/help/auths/locale/C/NetworkVRRP.html
1054 file path=usr/lib/help/auths/locale/C/PriAdmin.html
1055 file path=usr/lib/help/auths/locale/C/ProfmgrHeader.html
1056 file path=usr/lib/help/auths/locale/C/RoleHeader.html
1057 file path=usr/lib/help/auths/locale/C/SmfAllocate.html
1058 file path=usr/lib/help/auths/locale/C/SmfAutofsStates.html
1059 file path=usr/lib/help/auths/locale/C/SmfCoreadmStates.html
1060 file path=usr/lib/help/auths/locale/C/SmfCronStates.html
1061 file path=usr/lib/help/auths/locale/C/SmfExAcctFlowStates.html
1062 file path=usr/lib/help/auths/locale/C/SmfExAcctNetStates.html
1063 file path=usr/lib/help/auths/locale/C/SmfExAcctProcessStates.html
1064 file path=usr/lib/help/auths/locale/C/SmfExAcctTaskStates.html
1065 file path=usr/lib/help/auths/locale/C/SmfHeader.html
1066 file path=usr/lib/help/auths/locale/C/SmfILBStates.html
1067 file path=usr/lib/help/auths/locale/C/SmfIPsecStates.html
1068 file path=usr/lib/help/auths/locale/C/SmfIdmapStates.html
1069 file path=usr/lib/help/auths/locale/C/SmfInetdStates.html
1070 file path=usr/lib/help/auths/locale/C/SmfLocationStates.html
1071 file path=usr/lib/help/auths/locale/C/SmfMDNSStates.html
1072 file path=usr/lib/help/auths/locale/C/SmfManageAudit.html
1073 file path=usr/lib/help/auths/locale/C/SmfManageHeader.html
1074 file path=usr/lib/help/auths/locale/C/SmfManageHotplug.html
1075 file path=usr/lib/help/auths/locale/C/SmfManageZFSSnap.html
1076 file path=usr/lib/help/auths/locale/C/SmfModifyAppl.html
1077 file path=usr/lib/help/auths/locale/C/SmfModifyDepend.html
1078 file path=usr/lib/help/auths/locale/C/SmfModifyFramework.html
1079 file path=usr/lib/help/auths/locale/C/SmfModifyHeader.html
1080 file path=usr/lib/help/auths/locale/C/SmfModifyMethod.html
1081 file path=usr/lib/help/auths/locale/C/SmfNADDStates.html
1082 file path=usr/lib/help/auths/locale/C/SmfNDMPStates.html
1083 file path=usr/lib/help/auths/locale/C/SmfNWAMStates.html
1084 file path=usr/lib/help/auths/locale/C/SmfNscdStates.html
1085 file path=usr/lib/help/auths/locale/C/SmfPowerStates.html
1086 file path=usr/lib/help/auths/locale/C/SmfReparseStates.html
1087 file path=usr/lib/help/auths/locale/C/SmfRoutingStates.html
1088 file path=usr/lib/help/auths/locale/C/SmfSMBFSStates.html
1089 file path=usr/lib/help/auths/locale/C/SmfSMBStates.html
1090 file path=usr/lib/help/auths/locale/C/SmfSendmailStates.html
1091 file path=usr/lib/help/auths/locale/C/SmfSshStates.html
1092 file path=usr/lib/help/auths/locale/C/SmfSyslogStates.html
1093 file path=usr/lib/help/auths/locale/C/SmfVRRPStates.html
1094 file path=usr/lib/help/auths/locale/C/SmfValueAudit.html
1095 file path=usr/lib/help/auths/locale/C/SmfValueCoreadm.html
1096 file path=usr/lib/help/auths/locale/C/SmfValueExAcctFlow.html
1097 file path=usr/lib/help/auths/locale/C/SmfValueExAcctNet.html
1098 file path=usr/lib/help/auths/locale/C/SmfValueExAcctProcess.html
1099 file path=usr/lib/help/auths/locale/C/SmfValueExAcctTask.html
1100 file path=usr/lib/help/auths/locale/C/SmfValueFirewall.html
1101 file path=usr/lib/help/auths/locale/C/SmfValueHeader.html
1102 file path=usr/lib/help/auths/locale/C/SmfValueIPsec.html
1103 file path=usr/lib/help/auths/locale/C/SmfValueIdmap.html
1104 file path=usr/lib/help/auths/locale/C/SmfValueInetd.html
1105 file path=usr/lib/help/auths/locale/C/SmfValueMDNS.html
1106 file path=usr/lib/help/auths/locale/C/SmfValueNADD.html
1107 file path=usr/lib/help/auths/locale/C/SmfValueNDMP.html
1108 file path=usr/lib/help/auths/locale/C/SmfValueNWAM.html
1109 file path=usr/lib/help/auths/locale/C/SmfValueRouting.html
1110 file path=usr/lib/help/auths/locale/C/SmfValueSMB.html
1111 file path=usr/lib/help/auths/locale/C/SmfValueVscan.html
1112 file path=usr/lib/help/auths/locale/C/SmfValueVt.html
1113 file path=usr/lib/help/auths/locale/C/SmfVscanStates.html
1114 file path=usr/lib/help/auths/locale/C/SmfVtStates.html
1115 file path=usr/lib/help/auths/locale/C/SmfWpaStates.html
1116 file path=usr/lib/help/auths/locale/C/SysCpuPowerMgmt.html
```



```
1117 file path=usr/lib/help/auths/locale/C/SysDate.html
1118 file path=usr/lib/help/auths/locale/C/SysHeader.html
1119 file path=usr/lib/help/auths/locale/C/SysMaintenance.html
1120 file path=usr/lib/help/auths/locale/C/SysPowerMgmtBrightness.html
1121 file path=usr/lib/help/auths/locale/C/SysPowerMgmtHeader.html
1122 file path=usr/lib/help/auths/locale/C/SysPowerMgmtSuspend.html
1123 file path=usr/lib/help/auths/locale/C/SysPowerMgmtSuspendtoDisk.html
1124 file path=usr/lib/help/auths/locale/C/SysPowerMgmtSuspendtoRAM.html
1125 file path=usr/lib/help/auths/locale/C/SysShutdown.html
1126 file path=usr/lib/help/auths/locale/C/SysSyseventRead.html
1127 file path=usr/lib/help/auths/locale/C/SysSyseventWrite.html
1128 file path=usr/lib/help/auths/locale/C/WifiConfig.html
1129 file path=usr/lib/help/auths/locale/C/WifiWep.html
1128 file path=usr/lib/help/auths/locale/C/ZoneCloneFrom.html
1129 file path=usr/lib/help/auths/locale/C/ZoneHeader.html
1130 file path=usr/lib/help/auths/locale/C/ZoneLogin.html
1131 file path=usr/lib/help/auths/locale/C/ZoneManage.html
1132 file path=usr/lib/help/profiles/locale/C/RtAcctadm.html
1133 file path=usr/lib/help/profiles/locale/C/RtAll.html
1134 file path=usr/lib/help/profiles/locale/C/RtAuditCfg.html
1135 file path=usr/lib/help/profiles/locale/C/RtAuditCtrl.html
1136 file path=usr/lib/help/profiles/locale/C/RtAuditReview.html
1137 file path=usr/lib/help/profiles/locale/C/RtCPUPowerManagement.html
1138 file path=usr/lib/help/profiles/locale/C/RtConsUser.html
1139 file path=usr/lib/help/profiles/locale/C/RtContractObserver.html
1140 file path=usr/lib/help/profiles/locale/C/RtCronMngmnt.html
1141 file path=usr/lib/help/profiles/locale/C/RtCryptoMngmnt.html
1142 file path=usr/lib/help/profiles/locale/C/RtDHCPMngmnt.html
1143 file path=usr/lib/help/profiles/locale/C/RtDatAdmin.html
1144 file path=usr/lib/help/profiles/locale/C/RtDefault.html
1145 file path=usr/lib/help/profiles/locale/C/RtDeviceMngmnt.html
1146 file path=usr/lib/help/profiles/locale/C/RtDeviceSecurity.html
1147 file path=usr/lib/help/profiles/locale/C/RtExAcctFlow.html
1148 file path=usr/lib/help/profiles/locale/C/RtExAcctNet.html
1149 file path=usr/lib/help/profiles/locale/C/RtExAcctProcess.html
1150 file path=usr/lib/help/profiles/locale/C/RtExAcctTask.html
1151 file path=usr/lib/help/profiles/locale/C/RtFTPMngmnt.html
1152 file path=usr/lib/help/profiles/locale/C/RtFileSysMngmnt.html
1153 file path=usr/lib/help/profiles/locale/C/RtFileSysSecurity.html
1154 file path=usr/lib/help/profiles/locale/C/RtHotplugMngmnt.html
1155 file path=usr/lib/help/profiles/locale/C/RtIPFilterMngmnt.html
1156 file path=usr/lib/help/profiles/locale/C/RtIdmapMngmnt.html
1157 file path=usr/lib/help/profiles/locale/C/RtIdmapNameRulesMngmnt.html
1158 file path=usr/lib/help/profiles/locale/C/RtInetdMngmnt.html
1159 file path=usr/lib/help/profiles/locale/C/RtKerberosClntMngmnt.html
1160 file path=usr/lib/help/profiles/locale/C/RtKerberosSrvrMngmnt.html
1161 file path=usr/lib/help/profiles/locale/C/RtLogMngmnt.html
1162 file path=usr/lib/help/profiles/locale/C/RtMailMngmnt.html
1163 file path=usr/lib/help/profiles/locale/C/RtMaintAndRepair.html
1164 file path=usr/lib/help/profiles/locale/C/RtMediaBkup.html
1165 file path=usr/lib/help/profiles/locale/C/RtMediaCtlg.html
1166 file path=usr/lib/help/profiles/locale/C/RtMediaRestore.html
1167 file path=usr/lib/help/profiles/locale/C/RtNDMPMngmnt.html
1168 file path=usr/lib/help/profiles/locale/C/RtNameServiceAdmin.html
1169 file path=usr/lib/help/profiles/locale/C/RtNameServiceSecure.html
1170 file path=usr/lib/help/profiles/locale/C/RtNetAutoconfAdmin.html
1171 file path=usr/lib/help/profiles/locale/C/RtNetAutoconfUser.html
1172 file path=usr/lib/help/profiles/locale/C/RtNetILB.html
1173 file path=usr/lib/help/profiles/locale/C/RtNetIPsec.html
1174 file path=usr/lib/help/profiles/locale/C/RtNetLinkSecure.html
1175 file path=usr/lib/help/profiles/locale/C/RtNetMngmnt.html
1176 file path=usr/lib/help/profiles/locale/C/RtNetObservability.html
1177 file path=usr/lib/help/profiles/locale/C/RtNetSecure.html
1178 file path=usr/lib/help/profiles/locale/C/RtNetVRRP.html
1179 file path=usr/lib/help/profiles/locale/C/RtNetWifiMngmnt.html
1180 file path=usr/lib/help/profiles/locale/C/RtNetWifiSecure.html
```

```
1181 file path=usr/lib/help/profiles/locale/C/RtObAccessMngmnt.html
1182 file path=usr/lib/help/profiles/locale/C/RtOperator.html
1183 file path=usr/lib/help/profiles/locale/C/RtPriAdmin.html
1184 file path=usr/lib/help/profiles/locale/C/RtPrntAdmin.html
1185 file path=usr/lib/help/profiles/locale/C/RtProcManagement.html
1186 file path=usr/lib/help/profiles/locale/C/RtReparseMngmnt.html
1187 file path=usr/lib/help/profiles/locale/C/RtReservedProfile.html
1188 file path=usr/lib/help/profiles/locale/C/RtRightsDelegate.html
1189 file path=usr/lib/help/profiles/locale/C/RtSMBFSMngmnt.html
1190 file path=usr/lib/help/profiles/locale/C/RtSMBMngmnt.html
1191 file path=usr/lib/help/profiles/locale/C/RtSoftwareInstall.html
1192 file path=usr/lib/help/profiles/locale/C/RtSysAdmin.html
1193 file path=usr/lib/help/profiles/locale/C/RtSysEvMngmnt.html
1194 file path=usr/lib/help/profiles/locale/C/RtSysPowerMgmt.html
1195 file path=usr/lib/help/profiles/locale/C/RtSysPowerMgmtBrightness.html
1196 file path=usr/lib/help/profiles/locale/C/RtSysPowerMgmtSuspend.html
1197 file path=usr/lib/help/profiles/locale/C/RtSysPowerMgmtSuspendtoDisk.html
1198 file path=usr/lib/help/profiles/locale/C/RtSysPowerMgmtSuspendtoRAM.html
1199 file path=usr/lib/help/profiles/locale/C/RtUserMngmnt.html
1200 file path=usr/lib/help/profiles/locale/C/RtUserSecurity.html
1201 file path=usr/lib/help/profiles/locale/C/RtVscanMngmnt.html
1202 file path=usr/lib/help/profiles/locale/C/RtZFSFileSysMngmnt.html
1203 file path=usr/lib/help/profiles/locale/C/RtZFSStorageMngmnt.html
1204 file path=usr/lib/help/profiles/locale/C/RtZoneMngmnt.html
1205 file path=usr/lib/help/profiles/locale/C/RtZoneSecurity.html
1206 file path=usr/lib/hotplugd mode=0555
1207 file path=usr/lib/iconv/646da.8859.t mode=0444
1208 file path=usr/lib/iconv/646de.8859.t mode=0444
1209 file path=usr/lib/iconv/646en.8859.t mode=0444
1210 file path=usr/lib/iconv/646es.8859.t mode=0444
1211 file path=usr/lib/iconv/646fr.8859.t mode=0444
1212 file path=usr/lib/iconv/646it.8859.t mode=0444
1213 file path=usr/lib/iconv/646sv.8859.t mode=0444
1214 file path=usr/lib/iconv/8859.646.t mode=0444
1215 file path=usr/lib/iconv/8859.646da.t mode=0444
1216 file path=usr/lib/iconv/8859.646de.t mode=0444
1217 file path=usr/lib/iconv/8859.646en.t mode=0444
1218 file path=usr/lib/iconv/8859.646es.t mode=0444
1219 file path=usr/lib/iconv/8859.646fr.t mode=0444
1220 file path=usr/lib/iconv/8859.646it.t mode=0444
1221 file path=usr/lib/iconv/8859.646sv.t mode=0444
1222 file path=usr/lib/iconv/iconv_data mode=0444
1223 file path=usr/lib/idmapd mode=0555
1224 file path=usr/lib/inet/$(ARCH32)/in.iked mode=0555
1225 file path=usr/lib/inet/$(ARCH64)/in.iked mode=0555
1226 file path=usr/lib/inet/certdb mode=0555
1227 file path=usr/lib/inet/certlocal mode=0555
1228 file path=usr/lib/inet/certrldb mode=0555
1229 file path=usr/lib/inet/inetd mode=0555
1230 file path=usr/lib/intrd mode=0555
1231 file path=usr/lib/isaexec mode=0555
1232 file path=usr/lib/kssladm mode=0555
1233 $(sparc_ONLY)file path=usr/lib/ld.so
1234 file path=usr/lib/libshare.so.1
1235 file path=usr/lib/makekey mode=0555
1236 file path=usr/lib/more.help
1237 file path=usr/lib/newsyslog group=sys mode=0555
1238 file path=usr/lib/passmgmt group=sys mode=0555
1239 file path=usr/lib/pci/pcidr mode=0555
1240 file path=usr/lib/pci/pcidr_plugin.so
1241 file path=usr/lib/pfexecd mode=0555
1242 file path=usr/lib/platexec mode=0555
1243 file path=usr/lib/rcm/modules/SUNW_aggr_rcm.so mode=0555
1244 file path=usr/lib/rcm/modules/SUNW_cluster_rcm.so mode=0555
1245 file path=usr/lib/rcm/modules/SUNW_dump_rcm.so mode=0555
1246 file path=usr/lib/rcm/modules/SUNW_filesys_rcm.so mode=0555
```



```
1247 file path=usr/lib/rcm/modules/SUNW_ibpart_rcm.so mode=0555
1248 file path=usr/lib/rcm/modules/SUNW_ip_anon_rcm.so mode=0555
1249 file path=usr/lib/rcm/modules/SUNW_ip_rcm.so mode=0555
1250 file path=usr/lib/rcm/modules/SUNW_mpxio_rcm.so mode=0555
1251 file path=usr/lib/rcm/modules/SUNW_network_rcm.so mode=0555
1252 file path=usr/lib/rcm/modules/SUNW_swap_rcm.so mode=0555
1253 $(sparc_ONLY)file path=usr/lib/rcm/modules/SUNW_ttymux_rcm.so mode=0555
1254 file path=usr/lib/rcm/modules/SUNW_vlan_rcm.so mode=0555
1255 file path=usr/lib/rcm/modules/SUNW_vnic_rcm.so mode=0555
1256 file path=usr/lib/rcm/rcm_daemon mode=0555
1257 file path=usr/lib/reparse/reparsed group=sys mode=0555
1258 file path=usr/lib/saf/listen group=sys mode=0755
1259 file path=usr/lib/saf/nlps_server group=sys mode=0755
1260 file path=usr/lib/saf/sac group=sys mode=0555
1261 file path=usr/lib/saf/ttymon group=sys mode=0555
1262 file path=usr/lib/sysevent/modules/datalink_mod.so
1263 file path=usr/lib/sysevent/modules/devfsadmd_mod.so
1264 file path=usr/lib/sysevent/modules/sysevent_conf_mod.so
1265 file path=usr/lib/sysevent/modules/sysevent_reg_mod.so
1266 file path=usr/lib/sysevent/syseventconfd mode=0555
1267 file path=usr/lib/sysevent/syseventd mode=0555
1268 file path=usr/lib/utmp_update mode=4555
1269 file path=usr/lib/utmpd mode=0555
1270 file path=usr/lib/vtdaemon mode=0555
1271 file path=usr/lib/vtinfo mode=0555
1272 file path=usr/lib/vtxlock mode=0555
1273 file path=usr/sadm/bin/puttext mode=0555
1274 file path=usr/sadm/install/miniroot.db group=sys mode=0444
1275 file path=usr/sadm/install/scripts/i.ipsecalgs group=sys mode=0555
1276 file path=usr/sadm/install/scripts/i.kcfconf group=sys mode=0555
1277 file path=usr/sadm/install/scripts/i.kmfconf group=sys mode=0555
1278 file path=usr/sadm/install/scripts/i.manifest group=sys mode=0555
1279 file path=usr/sadm/install/scripts/i.pkcs11conf group=sys mode=0555
1280 file path=usr/sadm/install/scripts/i.rbac group=sys mode=0555
1281 file path=usr/sadm/install/scripts/r.ipsecalgs group=sys mode=0555
1282 file path=usr/sadm/install/scripts/r.kcfconf group=sys mode=0555
1283 file path=usr/sadm/install/scripts/r.kmfconf group=sys mode=0555
1284 file path=usr/sadm/install/scripts/r.manifest group=sys mode=0555
1285 file path=usr/sadm/install/scripts/r.pkcs11conf group=sys mode=0555
1286 file path=usr/sadm/install/scripts/r.rbac group=sys mode=0555
1287 file path=usr/sadm/ugdates mode=0444
1288 $(i386_ONLY)file path=usr/sbin/$(ARCH32)/add_drv group=sys mode=0555
1289 $(i386_ONLY)file path=usr/sbin/$(ARCH32)/modinfo group=sys mode=0555
1290 $(i386_ONLY)file path=usr/sbin/$(ARCH32)/modload group=sys mode=0555
1291 $(i386_ONLY)file path=usr/sbin/$(ARCH32)/modunload group=sys mode=0555
1292 $(i386_ONLY)file path=usr/sbin/$(ARCH32)/pbind group=sys mode=0555
1293 $(i386_ONLY)file path=usr/sbin/$(ARCH32)/prtconf group=sys mode=2555
1294 $(i386_ONLY)file path=usr/sbin/$(ARCH32)/psrset group=sys mode=0555
1295 $(i386_ONLY)file path=usr/sbin/$(ARCH32)/rem_drv group=sys mode=0555
1296 $(i386_ONLY)file path=usr/sbin/$(ARCH32)/swap group=sys mode=2555
1297 $(i386_ONLY)file path=usr/sbin/$(ARCH32)/sysdef group=sys mode=2555
1298 $(i386_ONLY)file path=usr/sbin/$(ARCH32)/update_drv group=sys mode=0555
1299 $(i386_ONLY)file path=usr/sbin/$(ARCH32)/whodo mode=4555
1300 file path=usr/sbin/$(ARCH64)/add_drv group=sys mode=0555
1301 file path=usr/sbin/$(ARCH64)/modinfo group=sys mode=0555
1302 file path=usr/sbin/$(ARCH64)/modload group=sys mode=0555
1303 file path=usr/sbin/$(ARCH64)/modunload group=sys mode=0555
1304 file path=usr/sbin/$(ARCH64)/pbind group=sys mode=0555
1305 file path=usr/sbin/$(ARCH64)/prtconf group=sys mode=2555
1306 file path=usr/sbin/$(ARCH64)/psrset group=sys mode=0555
1307 file path=usr/sbin/$(ARCH64)/rem_drv group=sys mode=0555
1308 file path=usr/sbin/$(ARCH64)/swap group=sys mode=2555
1309 file path=usr/sbin/$(ARCH64)/sysdef group=sys mode=2555
1310 file path=usr/sbin/$(ARCH64)/update_drv group=sys mode=0555
1311 file path=usr/sbin/$(ARCH64)/whodo mode=4555
1312 file path=usr/sbin/6to4relay mode=0555
```

```
1313 file path=usr/sbin/acctadm mode=0555
1314 file path=usr/sbin/allocate mode=4555
1315 file path=usr/sbin/arp mode=0555
1316 file path=usr/sbin/audit mode=0555
1317 file path=usr/sbin/auditconfig mode=0555
1318 file path=usr/sbin/auditd mode=0555
1319 file path=usr/sbin/auditrecord mode=0555
1320 file path=usr/sbin/auditreduce mode=0555
1321 file path=usr/sbin/auditstat mode=0555
1322 file path=usr/sbin/cfgadm mode=0555
1323 file path=usr/sbin/chroot mode=0555
1324 file path=usr/sbin/clear_locks mode=0555
1325 file path=usr/sbin/clinfo mode=0555
1326 file path=usr/sbin/clri mode=0555
1327 file path=usr/sbin/consadm group=sys mode=0555
1328 file path=usr/sbin/cron group=sys mode=0555
1329 file path=usr/sbin/devfsadm group=sys mode=0755
1330 file path=usr/sbin/devinfo mode=0555
1331 file path=usr/sbin/df mode=0555
1332 file path=usr/sbin/dfmounts mode=0555
1333 file path=usr/sbin/dispadmin mode=0555
1334 file path=usr/sbin/dminfo mode=0555
1335 file path=usr/sbin/dumpadm mode=0555
1336 file path=usr/sbin/eeprom group=sys mode=2555
1337 file path=usr/sbin/ff mode=0555
1338 file path=usr/sbin/fmthard group=sys mode=0555
1339 file path=usr/sbin/format mode=0555
1340 file path=usr/sbin/fsck mode=0555
1341 file path=usr/sbin/fstyp group=sys mode=0555
1342 file path=usr/sbin/fuser mode=0555
1343 file path=usr/sbin/getdevpolicy group=sys mode=0555
1344 file path=usr/sbin/getmajor group=sys mode=0755
1345 file path=usr/sbin/groupadd group=sys mode=0555
1346 file path=usr/sbin/groupdel group=sys mode=0555
1347 file path=usr/sbin/groupmod group=sys mode=0555
1348 file path=usr/sbin/grpck mode=0555
1349 file path=usr/sbin/halt mode=0755
1350 file path=usr/sbin/hotplug mode=0555
1351 file path=usr/sbin/idmap mode=0555
1352 file path=usr/sbin/if_mpadm mode=0555
1353 file path=usr/sbin/ikeadm mode=0555
1354 file path=usr/sbin/ikecert mode=0555
1355 file path=usr/sbin/inetadm mode=0555
1356 file path=usr/sbin/inetconv mode=0555
1357 file path=usr/sbin/install mode=0555
1358 file path=usr/sbin/installboot group=sys mode=0555
1359 file path=usr/sbin/ipaddrsel mode=0555
1360 file path=usr/sbin/ipsecalgs mode=0555
1361 file path=usr/sbin/ipsecconf mode=0555
1362 file path=usr/sbin/ipseckey mode=0555
1363 file path=usr/sbin/keyserv group=sys mode=0555
1364 file path=usr/sbin/killall mode=0555
1365 file path=usr/sbin/ksslcfg mode=0555
1366 file path=usr/sbin/link mode=0555
1367 file path=usr/sbin/locator mode=0555
1368 file path=usr/sbin/lofiadm mode=0555
1369 file path=usr/sbin/logadm mode=0555
1370 file path=usr/sbin/makedbm mode=0555
1371 file path=usr/sbin/mkdevalloc mode=0555
1372 file path=usr/sbin/mkfile mode=0555
1373 file path=usr/sbin/mknod mode=0555
1374 file path=usr/sbin/mountall group=sys mode=0555
1375 file path=usr/sbin/msgid mode=0555
1376 file path=usr/sbin/mvdir mode=0555
1377 file path=usr/sbin/ndd mode=0555
1378 file path=usr/sbin/nlsadmin group=adm mode=0755
```



```
1379 file path=usr/sbin/nscd mode=0555
1380 file path=usr/sbin/nwamadm mode=0555
1381 file path=usr/sbin/nwamcfg mode=0555
1382 file path=usr/sbin/pmadm group=sys mode=0555
1383 file path=usr/sbin/praudit mode=0555
1384 $(i386_ONLY)file path=usr/sbin/prtdiag group=sys mode=2755
1385 file path=usr/sbin/prtvtoc group=sys mode=0555
1386 file path=usr/sbin/psradm group=sys mode=0555
1387 file path=usr/sbin/psrinfo group=sys mode=0555
1388 file path=usr/sbin/pwck mode=0555
1389 file path=usr/sbin/pwconv group=sys mode=0555
1390 file path=usr/sbin/raidctl mode=0555
1391 file path=usr/sbin/ramdiskadm mode=0555
1392 file path=usr/sbin/rctladm mode=0555
1393 file path=usr/sbin/root_archive group=sys mode=0555
1394 file path=usr/sbin/rpcbind mode=0555
1395 $(i386_ONLY)file path=usr/sbin/rtc mode=0555
1396 file path=usr/sbin/sacadm group=sys mode=4755
1397 file path=usr/sbin/setmnt mode=0555
1398 file path=usr/sbin/shareall mode=0555
1399 file path=usr/sbin/sharectl mode=0555
1400 file path=usr/sbin/sharemgr mode=0555
1401 file path=usr/sbin/shutdown group=sys mode=0755
1402 file path=usr/sbin/smbios mode=0555
1403 file path=usr/sbin/stmsboot mode=0555
1404 file path=usr/sbin/strace group=sys mode=0555
1405 file path=usr/sbin/strclean group=sys mode=0555
1406 file path=usr/sbin/strerr group=sys mode=0555
1407 file path=usr/sbin/sttydefs group=sys mode=0755
1408 file path=usr/sbin/svcadm mode=0555
1409 file path=usr/sbin/svccfg mode=0555
1410 file path=usr/sbin/syncinit mode=0555
1411 file path=usr/sbin/syncloop mode=0555
1412 file path=usr/sbin/syncstat mode=0555
1413 file path=usr/sbin/syseventadm group=sys mode=0555
1414 file path=usr/sbin/syslogd group=sys mode=0555
1415 file path=usr/sbin/tar mode=0555
1416 file path=usr/sbin/traceroute mode=4555
1417 file path=usr/sbin/trapstat mode=0555
1418 file path=usr/sbin/ttyadm group=sys mode=0755
1419 $(i386_ONLY)file path=usr/sbin/ucodeadm mode=0555
1420 file path=usr/sbin/umountall group=sys mode=0555
1421 file path=usr/sbin/unlink mode=0555
1422 file path=usr/sbin/unshareall mode=0555
1423 file path=usr/sbin/useradd group=sys mode=0555
1424 file path=usr/sbin/userdel group=sys mode=0555
1425 file path=usr/sbin/usermod group=sys mode=0555
1426 $(sparc_ONLY)file path=usr/sbin/virtinfo mode=0555
1427 file path=usr/sbin/volcopy mode=0555
1428 file path=usr/sbin/wall group=tty mode=2555
1429 file path=usr/sbin/zdump mode=0555
1430 file path=usr/sbin/zic mode=0555
1431 file path=usr/share/doc/ksh/COMPATIBILITY
1432 file path=usr/share/doc/ksh/DESIGN
1433 file path=usr/share/doc/ksh/OBSOLETE
1434 file path=usr/share/doc/ksh/README
1435 file path=usr/share/doc/ksh/RELEASE
1436 file path=usr/share/doc/ksh/TYPES
1437 file path=usr/share/doc/ksh/images/callouts/1.png
1438 file path=usr/share/doc/ksh/images/callouts/10.png
1439 file path=usr/share/doc/ksh/images/callouts/2.png
1440 file path=usr/share/doc/ksh/images/callouts/3.png
1441 file path=usr/share/doc/ksh/images/callouts/4.png
1442 file path=usr/share/doc/ksh/images/callouts/5.png
1443 file path=usr/share/doc/ksh/images/callouts/6.png
1444 file path=usr/share/doc/ksh/images/callouts/7.png
```

```
1445 file path=usr/share/doc/ksh/images/callouts/8.png
1446 file path=usr/share/doc/ksh/images/callouts/9.png
1447 file path=usr/share/doc/ksh/images/tag_bourne.png
1448 file path=usr/share/doc/ksh/images/tag_i18n.png
1449 file path=usr/share/doc/ksh/images/tag_ksh.png
1450 file path=usr/share/doc/ksh/images/tag_ksh88.png
1451 file path=usr/share/doc/ksh/images/tag_ksh93.png
1452 file path=usr/share/doc/ksh/images/tag_l10n.png
1453 file path=usr/share/doc/ksh/images/tag_perf.png
1454 file path=usr/share/doc/ksh/shell_styleguide.docbook
1455 file path=usr/share/doc/ksh/shell_styleguide.html
1456 file path=usr/share/lib/mailx/mailx.help
1457 file path=usr/share/lib/mailx/mailx.help.~
1458 file path=usr/share/lib/tabset/3101
1459 file path=usr/share/lib/tabset/beehive
1460 file path=usr/share/lib/tabset/hds
1461 file path=usr/share/lib/tabset/hds3
1462 file path=usr/share/lib/tabset/std
1463 file path=usr/share/lib/tabset/stdcrt
1464 file path=usr/share/lib/tabset/teleray
1465 file path=usr/share/lib/tabset/vt100
1466 file path=usr/share/lib/tabset/wyse-adds
1467 file path=usr/share/lib/tabset/xerox1720
1468 file path=usr/share/lib/termcap
1469 file path=usr/share/lib/unittab
1470 file path=usr/share/lib/xml/dtd/adt_record.dtd.1
1471 file path=usr/share/lib/xml/dtd/kmfpolicy.dtd
1472 file path=usr/share/lib/xml/dtd/service_bundle.dtd.1 group=sys
1473 file path=usr/share/lib/xml/style/adt_record.xsl.1
1474 file path=var/adm/aculog mode=0600 owner=uucp preserve=true
1475 file path=var/adm/spellhist mode=0666 preserve=true
1476 file path=var/adm/utmpx preserve=true
1477 file path=var/adm/wtmpx group=adm owner=adm preserve=true
1478 file path=var/log/authlog group=sys mode=0600 preserve=true
1479 file path=var/log/syslog group=sys preserve=true
1480 file path=var/sadm/system/admin/default_java group=sys mode=0444
1481 file path=var/saf/zsmon/log group=sys preserve=true
1482 file path=var/spool/cron/crontabs/adm group=sys mode=0600 preserve=true
1483 file path=var/spool/cron/crontabs/root group=sys mode=0600 preserve=true
1484 hardlink path=etc/crypto/certs/SUNWObjectCA \
1485     target=../../../etc/certs/SUNWObjectCA
1486 hardlink path=etc/rc2.d/S20sysetup target=../../etc/init.d/sysetup
1487 hardlink path=etc/rc2.d/S73cachefs.daemon \
1488     target=../../etc/init.d/cachefs.daemon
1489 hardlink path=etc/rc2.d/S89PRESERVE target=../../etc/init.d/PRESERVE
1490 $(sparc_ONLY)hardlink path=etc/svc/profile/platform_SUNW,Sun-Fire-V890.xml \
1491     target=./platform_SUNW,Sun-Fire-880.xml
1492 $(sparc_ONLY)hardlink \
1493     path=etc/svc/profile/platform_SUNW,UltraSPARC-IIe-NetraCT-40.xml \
1494     target=./platform_SUNW,UltraSPARC-IIi-Netract.xml
1495 $(sparc_ONLY)hardlink \
1496     path=etc/svc/profile/platform_SUNW,UltraSPARC-IIe-NetraCT-60.xml \
1497     target=./platform_SUNW,UltraSPARC-IIi-Netract.xml
1498 hardlink path=sbin/rc5 target=../sbin/rc0
1499 hardlink path=sbin/rc6 target=../sbin/rc0
1500 hardlink path=usr/bin/$(ARCH32)/encrypt target=decrypt
1501 hardlink path=usr/bin/$(ARCH32)/ksh target=ksh93
1502 hardlink path=usr/bin/$(ARCH32)/mac target=digest
1503 hardlink path=usr/bin/$(ARCH32)/rksh target=ksh93
1504 hardlink path=usr/bin/$(ARCH32)/rksh93 target=ksh93
1505 $(i386_ONLY)hardlink path=usr/bin/$(ARCH32)/w target=uptime
1506 hardlink path=usr/bin/$(ARCH64)/encrypt target=decrypt
1507 hardlink path=usr/bin/$(ARCH64)/ksh target=ksh93
1508 hardlink path=usr/bin/$(ARCH64)/mac target=digest
1509 hardlink path=usr/bin/$(ARCH64)/rksh target=ksh93
1510 hardlink path=usr/bin/$(ARCH64)/rksh93 target=ksh93
```



```
1511 hardlink path=usr/bin/$(ARCH64)/w target=uptime
1512 hardlink path=usr/bin/bg target=../../usr/bin/alias
1513 hardlink path=usr/bin/cd target=../../usr/bin/alias
1514 hardlink path=usr/bin/cksum target=../../usr/bin/alias
1515 hardlink path=usr/bin/cmp target=../../usr/bin/alias
1516 hardlink path=usr/bin/comm target=../../usr/bin/alias
1517 hardlink path=usr/bin/command target=../../usr/bin/alias
1518 hardlink path=usr/bin/cut target=../../usr/bin/alias
1519 hardlink path=usr/bin/decrypt target=../../usr/lib/isaexec
1520 hardlink path=usr/bin/digest target=../../usr/lib/isaexec
1521 hardlink path=usr/bin/dispgid target=../../usr/bin/ckgid
1522 hardlink path=usr/bin/dispuid target=../../usr/bin/ckuid
1523 hardlink path=usr/bin/edit target=../has/bin/edit
1524 hardlink path=usr/bin/encrypt target=../../usr/lib/isaexec
1525 hardlink path=usr/bin/fc target=../../usr/bin/alias
1526 hardlink path=usr/bin/fg target=../../usr/bin/alias
1527 hardlink path=usr/bin/getopts target=../../usr/bin/alias
1528 hardlink path=usr/bin/hash target=../../usr/bin/alias
1529 hardlink path=usr/bin/i386 target=../../usr/bin/i286
1530 hardlink path=usr/bin/i486 target=../../usr/bin/i286
1531 hardlink path=usr/bin/i860 target=../../usr/bin/i286
1532 hardlink path=usr/bin/i86pc target=../../usr/bin/i286
1533 hardlink path=usr/bin/iAPX286 target=../../usr/bin/i286
1534 hardlink path=usr/bin/jobs target=../../usr/bin/alias
1535 hardlink path=usr/bin/join target=../../usr/bin/alias
1536 hardlink path=usr/bin/kill target=../../usr/bin/alias
1537 hardlink path=usr/bin/ksh target=../../usr/lib/isaexec
1538 hardlink path=usr/bin/ksh93 target=../../usr/lib/isaexec
1539 hardlink path=usr/bin/ln target=../../usr/bin/cp
1540 hardlink path=usr/bin/logname target=../../usr/bin/alias
1541 hardlink path=usr/bin/m68k target=../../usr/bin/i286
1542 hardlink path=usr/bin/mac target=../../usr/lib/isaexec
1543 hardlink path=usr/bin/mc68000 target=../../usr/bin/i286
1544 hardlink path=usr/bin/mc68010 target=../../usr/bin/i286
1545 hardlink path=usr/bin/mc68020 target=../../usr/bin/i286
1546 hardlink path=usr/bin/mc68030 target=../../usr/bin/i286
1547 hardlink path=usr/bin/mc68040 target=../../usr/bin/i286
1548 hardlink path=usr/bin/mv target=../../usr/bin/cp
1549 hardlink path=usr/bin/newtask target=../../usr/lib/isaexec
1550 hardlink path=usr/bin/nohup target=../../usr/lib/isaexec
1551 hardlink path=usr/bin/page target=../../usr/bin/more
1552 hardlink path=usr/bin/paste target=../../usr/bin/alias
1553 hardlink path=usr/bin/pdp11 target=../../usr/bin/i286
1554 hardlink path=usr/bin/pfbash target=../../usr/bin/pfexec
1555 hardlink path=usr/bin/pfcsh target=../../usr/bin/pfexec
1556 hardlink path=usr/bin/pfksh target=../../usr/bin/pfexec
1557 hardlink path=usr/bin/pfksh93 target=../../usr/bin/pfexec
1558 hardlink path=usr/bin/pfrksh target=../../usr/bin/pfexec
1559 hardlink path=usr/bin/pfrksh93 target=../../usr/bin/pfexec
1560 hardlink path=usr/bin/pfsh target=../../usr/bin/pfexec
1561 hardlink path=usr/bin/pftcsh target=../../usr/bin/pfexec
1562 hardlink path=usr/bin/pfzsh target=../../usr/bin/pfexec
1563 hardlink path=usr/bin/pkill target=../../usr/bin/pgrep
1564 hardlink path=usr/bin/prctl target=../../usr/lib/isaexec
1565 hardlink path=usr/bin/print target=../../usr/bin/alias
1566 hardlink path=usr/bin/prstat target=../../usr/lib/isaexec
1567 hardlink path=usr/bin/ps target=../../usr/lib/isaexec
1568 hardlink path=usr/bin/read target=../../usr/bin/alias
1569 hardlink path=usr/bin/red target=../../usr/bin/ed
1570 hardlink path=usr/bin/rev target=../../usr/bin/alias
1571 hardlink path=usr/bin/rksh target=../../usr/lib/isaexec
1572 hardlink path=usr/bin/rksh93 target=../../usr/lib/isaexec
1573 hardlink path=usr/bin/savecore target=../../usr/lib/isaexec
1574 hardlink path=usr/bin/setuname target=../../usr/lib/isaexec
1575 hardlink path=usr/bin/sleep target=../../usr/bin/alias
1576 hardlink path=usr/bin/sparc target=../../usr/bin/i286
```

```
1577 hardlink path=usr/bin/sum target=../../usr/bin/alias
1578 hardlink path=usr/bin/sun target=../../usr/bin/i286
1579 hardlink path=usr/bin/sun2 target=../../usr/bin/i286
1580 hardlink path=usr/bin/sun3 target=../../usr/bin/i286
1581 hardlink path=usr/bin/sun3x target=../../usr/bin/i286
1582 hardlink path=usr/bin/sun4 target=../../usr/bin/i286
1583 hardlink path=usr/bin/sun4c target=../../usr/bin/i286
1584 hardlink path=usr/bin/sun4d target=../../usr/bin/i286
1585 hardlink path=usr/bin/sun4e target=../../usr/bin/i286
1586 hardlink path=usr/bin/sun4m target=../../usr/bin/i286
1587 hardlink path=usr/bin/tee target=../../usr/bin/alias
1588 hardlink path=usr/bin/test target=../../usr/bin/alias
1589 hardlink path=usr/bin/touch target=../../usr/bin/settime
1590 hardlink path=usr/bin/type target=../../usr/bin/alias
1591 hardlink path=usr/bin/u370 target=../../usr/bin/i286
1592 hardlink path=usr/bin/u3b target=../../usr/bin/i286
1593 hardlink path=usr/bin/u3b15 target=../../usr/bin/i286
1594 hardlink path=usr/bin/u3b2 target=../../usr/bin/i286
1595 hardlink path=usr/bin/u3b5 target=../../usr/bin/i286
1596 hardlink path=usr/bin/ulimit target=../../usr/bin/alias
1597 hardlink path=usr/bin/umask target=../../usr/bin/alias
1598 hardlink path=usr/bin/unalias target=../../usr/bin/alias
1599 hardlink path=usr/bin/uniq target=../../usr/bin/alias
1600 hardlink path=usr/bin/uptime target=../../usr/lib/isaexec
1601 hardlink path=usr/bin/vax target=../../usr/bin/i286
1602 hardlink path=usr/bin/vedit target=../has/bin/edit
1603 hardlink path=usr/bin/w target=../../usr/lib/isaexec
1604 hardlink path=usr/bin/wait target=../../usr/bin/alias
1605 hardlink path=usr/bin/wc target=../../usr/bin/alias
1606 hardlink path=usr/has/bin/ex target=edit
1607 hardlink path=usr/has/bin/pfsh target=../../bin/pfexec
1608 hardlink path=usr/has/bin/vedit target=edit
1609 hardlink path=usr/has/bin/vi target=edit
1610 hardlink path=usr/has/bin/view target=edit
1611 hardlink path=usr/lib/fs/hsfs/fstyp target=../../../sbin/fstyp
1612 hardlink path=usr/lib/fs/ufs/dcopy target=../../../../usr/lib/fs/ufs/clri
1613 hardlink path=usr/lib/fs/ufs/fstyp target=../../../sbin/fstyp
1614 hardlink path=usr/lib/fs/ufs/quotaon \
1615     target=../../../../usr/lib/fs/ufs/quotaoff
1616 hardlink path=usr/lib/inet/in.iked target=../../../usr/lib/isaexec
1617 hardlink path=usr/sadm/bin/dispgid target=../../../usr/bin/ckgid
1618 hardlink path=usr/sadm/bin/dispuid target=../../../usr/bin/ckuid
1619 hardlink path=usr/sadm/bin/errange target=../../../usr/bin/ckrange
1620 hardlink path=usr/sadm/bin/errdate target=../../../usr/bin/ckdate
1621 hardlink path=usr/sadm/bin/errgid target=../../../usr/bin/ckgid
1622 hardlink path=usr/sadm/bin/errint target=../../../usr/bin/ckint
1623 hardlink path=usr/sadm/bin/erritem target=../../../usr/bin/ckitem
1624 hardlink path=usr/sadm/bin/errpath target=../../../usr/bin/ckpath
1625 hardlink path=usr/sadm/bin/errstr target=../../../usr/bin/ckstr
1626 hardlink path=usr/sadm/bin/errtime target=../../../usr/bin/cktime
1627 hardlink path=usr/sadm/bin/erruid target=../../../usr/bin/ckuid
1628 hardlink path=usr/sadm/bin/erryorn target=../../../usr/bin/ckyorn
1629 hardlink path=usr/sadm/bin/helpdate target=../../../usr/bin/ckdate
1630 hardlink path=usr/sadm/bin/helpgid target=../../../usr/bin/ckgid
1631 hardlink path=usr/sadm/bin/helpint target=../../../usr/bin/ckint
1632 hardlink path=usr/sadm/bin/helpitem target=../../../usr/bin/ckitem
1633 hardlink path=usr/sadm/bin/helppath target=../../../usr/bin/ckpath
1634 hardlink path=usr/sadm/bin/helprange target=../../../usr/bin/ckrange
1635 hardlink path=usr/sadm/bin/helpstr target=../../../usr/bin/ckstr
1636 hardlink path=usr/sadm/bin/helptime target=../../../usr/bin/cktime
1637 hardlink path=usr/sadm/bin/helpuid target=../../../usr/bin/ckuid
1638 hardlink path=usr/sadm/bin/helpyorn target=../../../usr/bin/ckyorn
1639 hardlink path=usr/sadm/bin/valdate target=../../../usr/bin/ckdate
1640 hardlink path=usr/sadm/bin/valgid target=../../../usr/bin/ckgid
1641 hardlink path=usr/sadm/bin/valint target=../../../usr/bin/ckint
1642 hardlink path=usr/sadm/bin/valpath target=../../../usr/bin/ckpath
```



```
1643 hardlink path=usr/sadm/bin/valrange target=../../../usr/bin/ckrange
1644 hardlink path=usr/sadm/bin/valstr target=../../../usr/bin/ckstr
1645 hardlink path=usr/sadm/bin/valtime target=../../../usr/bin/cktime
1646 hardlink path=usr/sadm/bin/valuid target=../../../usr/bin/ckuid
1647 hardlink path=usr/sadm/bin/valyorn target=../../../usr/bin/ckyorn
1648 hardlink path=usr/sbin/add_drv target=../../usr/lib/isaexec
1649 hardlink path=usr/sbin/audlinks target=./devfsadm
1650 hardlink path=usr/sbin/consadmd target=../../usr/sbin/consadm
1651 hardlink path=usr/sbin/deallocate target=../../usr/sbin/allocate
1652 hardlink path=usr/sbin/devlinks target=./devfsadm
1653 hardlink path=usr/sbin/dfshares target=../../usr/sbin/dfmounts
1654 hardlink path=usr/sbin/disks target=./devfsadm
1655 hardlink path=usr/sbin/drvconfig target=./devfsadm
1656 hardlink path=usr/sbin/list_devices target=../../usr/sbin/allocate
1657 hardlink path=usr/sbin/mkdevmaps target=../../usr/sbin/mkdevalloc
1658 hardlink path=usr/sbin/modinfo target=../../usr/lib/isaexec
1659 hardlink path=usr/sbin/modload target=../../usr/lib/isaexec
1660 hardlink path=usr/sbin/modunload target=../../usr/lib/isaexec
1661 hardlink path=usr/sbin/pbind target=../../usr/lib/isaexec
1662 hardlink path=usr/sbin/ports target=./devfsadm
1663 hardlink path=usr/sbin/poweroff target=./halt
1664 hardlink path=usr/sbin/prtconf target=../../usr/lib/isaexec
1665 $(sparc_ONLY)hardlink path=usr/sbin/prtdiag target=../../usr/lib/platexec
1666 hardlink path=usr/sbin/psrset target=../../usr/lib/isaexec
1667 hardlink path=usr/sbin/reboot target=./halt
1668 hardlink path=usr/sbin/rem_drv target=../../usr/lib/isaexec
1669 hardlink path=usr/sbin/roleadd target=../../usr/sbin/useradd
1670 hardlink path=usr/sbin/roledel target=../../usr/sbin/userdel
1671 hardlink path=usr/sbin/rolemod target=../../usr/sbin/usermod
1672 hardlink path=usr/sbin/share target=../../usr/sbin/sharemgr
1673 hardlink path=usr/sbin/swap target=../../usr/lib/isaexec
1674 hardlink path=usr/sbin/sysdef target=../../usr/lib/isaexec
1675 hardlink path=usr/sbin/tapes target=./devfsadm
1676 hardlink path=usr/sbin/unshare target=../../usr/sbin/sharemgr
1677 hardlink path=usr/sbin/update_drv target=../../usr/lib/isaexec
1678 hardlink path=usr/sbin/whodo target=../../usr/lib/isaexec
1679 legacy pkg=SUNWcsr \
1680     desc="core software for a specific instruction-set architecture" \
1681     name="Core Solaris, (Root)"
1682 legacy pkg=SUNWcsu \
1683     desc="core software for a specific instruction-set architecture" \
1684     name="Core Solaris, (Usr)"
1685 legacy pkg=SUNWftpr desc="FTP Server Configuration Files" \
1686     name="FTP Server, (Root)"
1687 license cr_Sun license=cr_Sun
1688 license lic_CDDL license=lic_CDDL
1689 license usr/src/cmd/cmd-inet/sbin/ifparse/THIRDPARTYLICENSE \
1690     license=usr/src/cmd/cmd-inet/sbin/ifparse/THIRDPARTYLICENSE
1691 license usr/src/cmd/cmd-inet/usr.lib/in.mpathd/THIRDPARTYLICENSE \
1692     license=usr/src/cmd/cmd-inet/usr.lib/in.mpathd/THIRDPARTYLICENSE
1693 license usr/src/cmd/cmd-inet/usr.sbin/THIRDPARTYLICENSE.arp \
1694     license=usr/src/cmd/cmd-inet/usr.sbin/THIRDPARTYLICENSE.arp
1695 license usr/src/cmd/cmd-inet/usr.sbin/THIRDPARTYLICENSE.route \
1696     license=usr/src/cmd/cmd-inet/usr.sbin/THIRDPARTYLICENSE.route
1697 license usr/src/cmd/cmd-inet/usr.sbin/ifconfig/THIRDPARTYLICENSE \
1698     license=usr/src/cmd/cmd-inet/usr.sbin/ifconfig/THIRDPARTYLICENSE
1699 license usr/src/cmd/cmd-inet/usr.sbin/in.ftpd/LICENSE \
1700     license=usr/src/cmd/cmd-inet/usr.sbin/in.ftpd/LICENSE
1701 license usr/src/cmd/cmd-inet/usr.sbin/traceroute/THIRDPARTYLICENSE \
1702     license=usr/src/cmd/cmd-inet/usr.sbin/traceroute/THIRDPARTYLICENSE
1703 license usr/src/cmd/cron/THIRDPARTYLICENSE \
1704     license=usr/src/cmd/cron/THIRDPARTYLICENSE
1705 license usr/src/cmd/csh/THIRDPARTYLICENSE \
1706     license=usr/src/cmd/csh/THIRDPARTYLICENSE
1707 license usr/src/cmd/eeprom/THIRDPARTYLICENSE \
1708     license=usr/src/cmd/eeprom/THIRDPARTYLICENSE
```

```
1709 license usr/src/cmd/fs.d/ufs/THIRDPARTYLICENSE \
1710     license=usr/src/cmd/fs.d/ufs/THIRDPARTYLICENSE
1711 license usr/src/cmd/mt/THIRDPARTYLICENSE \
1712     license=usr/src/cmd/mt/THIRDPARTYLICENSE
1713 license usr/src/cmd/script/THIRDPARTYLICENSE \
1714     license=usr/src/cmd/script/THIRDPARTYLICENSE
1715 license usr/src/cmd/sed/THIRDPARTYLICENSE \
1716     license=usr/src/cmd/sed/THIRDPARTYLICENSE
1717 license usr/src/cmd/stat/vmstat/THIRDPARTYLICENSE \
1718     license=usr/src/cmd/stat/vmstat/THIRDPARTYLICENSE
1719 license usr/src/cmd/tail/THIRDPARTYLICENSE \
1720     license=usr/src/cmd/tail/THIRDPARTYLICENSE
1721 license usr/src/cmd/tip/THIRDPARTYLICENSE \
1722     license=usr/src/cmd/tip/THIRDPARTYLICENSE
1723 license usr/src/cmd/tr/THIRDPARTYLICENSE \
1724     license=usr/src/cmd/tr/THIRDPARTYLICENSE
1725 license usr/src/cmd/vi/THIRDPARTYLICENSE \
1726     license=usr/src/cmd/vi/THIRDPARTYLICENSE
1727 license usr/src/cmd/which/THIRDPARTYLICENSE \
1728     license=usr/src/cmd/which/THIRDPARTYLICENSE
1729 license usr/src/cmd/xstr/THIRDPARTYLICENSE \
1730     license=usr/src/cmd/xstr/THIRDPARTYLICENSE
1731 license usr/src/common/bzip2/LICENSE license=usr/src/common/bzip2/LICENSE
1732 link path=bin target=./usr/bin
1733 link path=etc/TIMEZONE target=./default/init
1734 link path=etc/autopush target=../sbin/autopush
1735 link path=etc/cfgadm target=../usr/sbin/cfgadm
1736 link path=etc/clri target=../usr/sbin/clri
1737 link path=etc/cron target=../usr/sbin/cron
1738 link path=etc/dcopy target=../usr/sbin/dcopy
1739 link path=etc/ff target=../usr/sbin/ff
1740 link path=etc/fmthard target=../usr/sbin/fmthard
1741 link path=etc/format target=../usr/sbin/format
1742 link path=etc/fsck target=../usr/sbin/fsck
1743 link path=etc/fsdb target=../usr/sbin/fsdb
1744 link path=etc/fstyp target=../usr/sbin/fstyp
1745 link path=etc/getty target=../usr/lib/saf/ttymon
1746 link path=etc/grpck target=../usr/sbin/grpck
1747 link path=etc/halt target=../usr/sbin/halt
1748 link path=etc/hosts target=./inet/hosts
1749 link path=etc/inet/ipnodes target=./hosts
1750 link path=etc/inetd.conf target=./inet/inetd.conf
1751 link path=etc/init target=../sbin/init
1752 link path=etc/install target=../usr/sbin/install
1753 link path=etc/killall target=../usr/sbin/killall
1754 link path=etc/labelit target=../usr/sbin/labelit
1755 link path=etc/lib/ld.so.1 target=../../lib/ld.so.1
1756 link path=etc/lib/libdl.so.1 target=../../lib/libdl.so.1
1757 link path=etc/lib/nss_files.so.1 target=../../lib/nss_files.so.1
1758 link path=etc/log target=../var/adm/log
1759 link path=etc/mkfs target=../usr/sbin/mkfs
1760 link path=etc/mknod target=../usr/sbin/mknod
1761 link path=etc/mount target=../sbin/mount
1762 link path=etc/mountall target=../sbin/mountall
1763 link path=etc/ncheck target=../usr/sbin/ncheck
1764 link path=etc/netmasks target=./inet/netmasks
1765 link path=etc/networks target=./inet/networks
1766 link path=etc/protocols target=./inet/protocols
1767 link path=etc/prtconf target=../usr/sbin/prtconf
1768 link path=etc/prtvtoc target=../usr/sbin/prtvtoc
1769 link path=etc/rc0 target=../sbin/rc0
1770 link path=etc/rc1 target=../sbin/rc1
1771 link path=etc/rc2 target=../sbin/rc2
1772 link path=etc/rc3 target=../sbin/rc3
1773 link path=etc/rc5 target=../sbin/rc5
1774 link path=etc/rc6 target=../sbin/rc6
```



```
1775 link path=etc/rcS target=../sbin/rcS
1776 link path=etc/reboot target=../usr/sbin/halt
1777 link path=etc/security/audit/localhost/files target=../../../../var/audit
1778 link path=etc/services target=./inet/services
1779 link path=etc/setmnt target=../usr/sbin/setmnt
1780 link path=etc/shutdown target=../usr/sbin/shutdown
1781 link path=etc/sulogin target=../sbin/sulogin
1782 link path=etc/swap target=../usr/sbin/swap
1783 link path=etc/swapadd target=../sbin/swapadd
1784 link path=etc/sysdef target=../usr/sbin/sysdef
1785 link path=etc/tar target=../usr/sbin/tar
1786 link path=etc/telinit target=../sbin/init
1787 link path=etc/uadmin target=../sbin/uadmin
1788 link path=etc/umount target=../sbin/umount
1789 link path=etc/umountall target=../sbin/umountall
1790 link path=etc/utmpx target=../var/adm/utmpx
1791 link path=etc/volcopy target=../usr/sbin/volcopy
1792 link path=etc/wall target=../usr/sbin/wall
1793 link path=etc/whodo target=../usr/sbin/whodo
1794 link path=etc/wtmpx target=../var/adm/wtmpx
1795 link path=sbin/in.mpathd target=../lib/inet/in.mpathd
1796 link path=sbin/jsh target=../usr/bin/ksh93
1797 link path=sbin/pfsh target=../usr/bin/pfexec
1798 link path=sbin/sh target=../usr/bin/$(ARCH32)/ksh93
1799 link path=sbin/su target=../usr/bin/su
1800 link path=usr/adm target=../var/adm
1801 link path=usr/bin/cachefspack target=../lib/fs/cachefs/cachefspack
1802 link path=usr/bin/cachefsstat target=../lib/fs/cachefs/cachefsstat
1803 link path=usr/bin/df target=../sbin/df
1804 link path=usr/bin/jsh target=ksh93
1805 link path=usr/bin/pwconv target=../sbin/pwconv
1806 link path=usr/bin/rmail target=./mail
1807 link path=usr/bin/sh target=$(ARCH32)/ksh93
1808 link path=usr/bin/strclean target=../sbin/strclean
1809 link path=usr/bin/strerr target=../sbin/strerr
1810 link path=usr/bin/sync target=../../sbin/sync
1811 link path=usr/bin/tar target=../sbin/tar
1812 link path=usr/bin/uname target=../../sbin/uname
1813 link path=usr/ccs/bin/m4 target=../../bin/m4
1814 link path=usr/has/bin/jsh target=sh
1815 link path=usr/has/lib/rsh target=../bin/sh
1816 link path=usr/lib/$(ARCH64)/ld.so.1 target=../../../lib/$(ARCH64)/ld.so.1
1817 link path=usr/lib/cron target=../../etc/cron.d
1818 link path=usr/lib/devfsadm/devfsadmd target=../../sbin/devfsadm
1819 link path=usr/lib/embedded_su target=../bin/su
1820 link path=usr/lib/fs/dev/mount target=../../../../etc/fs/dev/mount
1821 link path=usr/lib/fs/hsfs/mount target=../../../../etc/fs/hsfs/mount
1822 link path=usr/lib/fs/ufs/mount target=../../../../etc/fs/ufs/mount
1823 link path=usr/lib/inet/in.mpathd target=../../../lib/inet/in.mpathd
1824 link path=usr/lib/ld.so.1 target=../../lib/ld.so.1
1825 link path=usr/lib/locale/POSIX target=./C
1826 link path=usr/lib/rsh target=../bin/ksh93
1827 link path=usr/lib/secure/32 target=.
1828 link path=usr/lib/secure/64 target=$(ARCH64)
1829 link path=usr/lib/wusbd target=../../sbin/wusbadm
1830 link path=usr/mail target=../var/mail
1831 link path=usr/net/nls/listen target=../../lib/saf/listen
1832 link path=usr/net/nls/nlps_server target=../../lib/saf/nlps_server
1833 link path=usr/news target=../var/news
1834 link path=usr/preserve target=../var/preserve
1835 link path=usr/pub target=./share/lib/pub
1836 link path=usr/sbin/autopush target=../../sbin/autopush
1837 link path=usr/sbin/bootadm target=../../sbin/bootadm
1838 link path=usr/sbin/cachefslog target=../lib/fs/cachefs/cachefslog
1839 link path=usr/sbin/cachefswssize target=../lib/fs/cachefs/cachefswssize
1840 link path=usr/sbin/cfsadmin target=../lib/fs/cachefs/cfsadmin
```

```
1841 link path=usr/sbin/cryptoadm target=../../sbin/cryptoadm
1842 link path=usr/sbin/dcopy target=./clri
1843 link path=usr/sbin/devnm target=./df
1844 link path=usr/sbin/dladm target=../../sbin/dladm
1845 link path=usr/sbin/dlstat target=../../sbin/dlstat
1846 link path=usr/sbin/edquota target=../lib/fs/ufs/edquota
1847 link path=usr/sbin/fdisk target=../../sbin/fdisk
1848 link path=usr/sbin/fiocompress target=../../sbin/fiocompress
1849 link path=usr/sbin/flowadm target=../../sbin/flowadm
1850 link path=usr/sbin/flowstat target=../../sbin/flowstat
1851 link path=usr/sbin/fsdb target=./clri
1852 link path=usr/sbin/fsirand target=../lib/fs/ufs/fsirand
1853 link path=usr/sbin/fssnap target=./clri
1854 link path=usr/sbin/hostconfig target=../../sbin/hostconfig
1855 link path=usr/sbin/ifconfig target=../../sbin/ifconfig
1856 link path=usr/sbin/inetd target=../lib/inet/inetd
1857 link path=usr/sbin/init target=../../sbin/init
1858 $(i386_ONLY)link path=usr/sbin/installgrub target=../../sbin/installgrub
1859 link path=usr/sbin/ipadm target=../../sbin/ipadm
1860 link path=usr/sbin/ipmpstat target=../../sbin/ipmpstat
1861 link path=usr/sbin/labelit target=./clri
1862 link path=usr/sbin/lockfs target=../lib/fs/ufs/lockfs
1863 link path=usr/sbin/mkfs target=./clri
1864 link path=usr/sbin/mount target=../../sbin/mount
1865 link path=usr/sbin/ncheck target=./ff
1866 link path=usr/sbin/newfs target=../lib/fs/ufs/newfs
1867 link path=usr/sbin/quot target=../lib/fs/ufs/quot
1868 link path=usr/sbin/quota target=../lib/fs/ufs/quota
1869 link path=usr/sbin/quotacheck target=../lib/fs/ufs/quotacheck
1870 link path=usr/sbin/quotaoff target=../lib/fs/ufs/quotaoff
1871 link path=usr/sbin/quotaon target=../lib/fs/ufs/quotaon
1872 link path=usr/sbin/repquota target=../lib/fs/ufs/repquota
1873 link path=usr/sbin/route target=../../sbin/route
1874 link path=usr/sbin/routeadm target=../../sbin/routeadm
1875 link path=usr/sbin/sync target=../../sbin/sync
1876 link path=usr/sbin/tunefs target=../lib/fs/ufs/tunefs
1877 link path=usr/sbin/tzreload target=../../sbin/tzreload
1878 link path=usr/sbin/uadmin target=../../sbin/uadmin
1879 link path=usr/sbin/ufsdump target=../lib/fs/ufs/ufsdump
1880 link path=usr/sbin/ufsrestore target=../lib/fs/ufs/ufsrestore
1881 link path=usr/sbin/umount target=../../sbin/umount
1882 link path=usr/sbin/wusbadm target=../../sbin/wusbadm
1883 link path=usr/spool target=../var/spool
1884 link path=usr/src target=./share/src
1885 link path=usr/tmp target=../var/tmp
1886 link path=var/ld/32 target=.
1887 link path=var/ld/64 target=$(ARCH64)
1888 #
1889 # The bootadm binary needs the etc/release file.
1890 #
1891 depend fmri=release/name type=require
1892 #
1893 # intrd and others use the illumos-defaulted perl interpreter
1894 #
1895 depend fmri=runtime/perl-510 type=require
1896 #
1897 # The loadkeys binary needs the keytables.
1898 #
1899 depend fmri=system/data/keyboard/keytables type=require
1900 #
1901 # Depend on terminfo data.
1902 #
1903 depend fmri=system/data/terminfo type=require
1904 #
1905 # Depend on zoneinfo data.
1906 #
```



```
1907 depend fmri=system/data/zoneinfo type=require
```

```
**********************************************************
   24012 Tue Jun 12 19:55:13 2012
new/usr/src/pkg/manifests/consolidation-osnet-osnet-message-files.mf
removed wificonfig tool
are /dev/wifi/* devices links now deprecated?
**********************************************************
   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 # Copyright (c) 2010, Oracle and/or its affiliates. All rights reserved.

  24 set name=pkg.fmri \
  25     value=pkg:/consolidation/osnet/osnet-message-files@$(PKGVERS)
  26 set name=pkg.description \
  27     value="localizable message files for the OS-Networking consolidation"
  28 set name=pkg.summary value="Localizable ON message files"
  29 set name=info.classification \
  30     value=org.opensolaris.category.2008:Development/System

  32 #
  33 # This package should not have automated dependencies generated because
  34 # it provides messages only.
  35 #
  36 set name=org.opensolaris.nodepend value=true
  37 set name=variant.arch value=$(ARCH)
  38 dir path=usr group=sys
  39 dir path=usr/lib
  40 dir path=usr/lib/help
  41 dir path=usr/lib/help/auths
  42 dir path=usr/lib/help/auths/locale
  43 dir path=usr/lib/help/profiles
  44 dir path=usr/lib/help/profiles/locale
  45 dir path=usr/lib/locale
  46 dir path=usr/lib/locale/C
  47 dir path=usr/lib/locale/C/LC_MESSAGES
  48 dir path=usr/lib/locale/C/LC_TIME
  49 dir path=usr/share
  50 dir path=usr/share/lib
  51 dir path=usr/share/lib/locale
  52 dir path=usr/share/lib/locale/com
  53 dir path=usr/share/lib/locale/com/sun
  54 dir path=usr/share/lib/locale/com/sun/admin
  55 dir path=usr/share/lib/locale/com/sun/admin/pm
  56 dir path=usr/share/lib/locale/com/sun/admin/pm/client
  57 dir path=usr/share/lib/locale/com/sun/dhcpmgr
  58 dir path=usr/share/lib/locale/com/sun/dhcpmgr/bridge
  59 dir path=usr/share/lib/locale/com/sun/dhcpmgr/cli
  60 dir path=usr/share/lib/locale/com/sun/dhcpmgr/cli/common
```



```
  61 dir path=usr/share/lib/locale/com/sun/dhcpmgr/cli/dhcpbatch
  62 dir path=usr/share/lib/locale/com/sun/dhcpmgr/cli/dhcpconfig
  63 dir path=usr/share/lib/locale/com/sun/dhcpmgr/cli/dhtadm
  64 dir path=usr/share/lib/locale/com/sun/dhcpmgr/cli/pntadm
  65 dir path=usr/share/lib/locale/com/sun/dhcpmgr/client
  66 dir path=usr/share/lib/locale/com/sun/dhcpmgr/client/SUNWbinfiles
  67 dir path=usr/share/lib/locale/com/sun/dhcpmgr/client/SUNWfiles
  68 dir path=usr/share/lib/locale/com/sun/dhcpmgr/client/help
  69 dir path=usr/share/lib/locale/com/sun/dhcpmgr/client/help/art
  70 dir path=usr/share/lib/locale/com/sun/dhcpmgr/common
  71 dir path=usr/share/lib/locale/com/sun/dhcpmgr/data
  72 dir path=usr/share/lib/locale/com/sun/dhcpmgr/ui
  73 dir path=usr/share/lib/locale/com/sun/slp
  74 file path=usr/lib/help/auths/locale/AllSolAuthsHeader.html
  75 file path=usr/lib/help/auths/locale/AuditHeader.html
  76 file path=usr/lib/help/auths/locale/AuthJobsAdmin.html
  77 file path=usr/lib/help/auths/locale/AuthJobsUser.html
  78 file path=usr/lib/help/auths/locale/AuthProfmgrAssign.html
  79 file path=usr/lib/help/auths/locale/AuthProfmgrDelegate.html
  80 file path=usr/lib/help/auths/locale/AuthProfmgrExecattrWrite.html
  81 file path=usr/lib/help/auths/locale/AuthProfmgrRead.html
  82 file path=usr/lib/help/auths/locale/AuthProfmgrWrite.html
  83 file path=usr/lib/help/auths/locale/AuthReadNDMP.html
  84 file path=usr/lib/help/auths/locale/AuthReadSMB.html
  85 file path=usr/lib/help/auths/locale/AuthRoleAssign.html
  86 file path=usr/lib/help/auths/locale/AuthRoleDelegate.html
  87 file path=usr/lib/help/auths/locale/AuthRoleWrite.html
  88 file path=usr/lib/help/auths/locale/BindStates.html
  89 file path=usr/lib/help/auths/locale/DevAllocHeader.html
  90 file path=usr/lib/help/auths/locale/DevAllocate.html
  91 file path=usr/lib/help/auths/locale/DevCDRW.html
  92 file path=usr/lib/help/auths/locale/DevConfig.html
  93 file path=usr/lib/help/auths/locale/DevGrant.html
  94 file path=usr/lib/help/auths/locale/DevRevoke.html
  95 file path=usr/lib/help/auths/locale/DhcpmgrHeader.html
  96 file path=usr/lib/help/auths/locale/DhcpmgrWrite.html
  97 file path=usr/lib/help/auths/locale/FileChown.html
  98 file path=usr/lib/help/auths/locale/FileHeader.html
  99 file path=usr/lib/help/auths/locale/FileOwner.html
 100 file path=usr/lib/help/auths/locale/HotplugHeader.html
 101 file path=usr/lib/help/auths/locale/HotplugModify.html
 102 file path=usr/lib/help/auths/locale/IdmapRules.html
 103 file path=usr/lib/help/auths/locale/JobHeader.html
 104 file path=usr/lib/help/auths/locale/JobsGrant.html
 105 file path=usr/lib/help/auths/locale/LabelFileDowngrade.html
 106 file path=usr/lib/help/auths/locale/LabelFileUpgrade.html
 107 file path=usr/lib/help/auths/locale/LabelHeader.html
 108 file path=usr/lib/help/auths/locale/LabelPrint.html
 109 file path=usr/lib/help/auths/locale/LabelRange.html
 110 file path=usr/lib/help/auths/locale/LabelServer.html
 111 file path=usr/lib/help/auths/locale/LabelWinDowngrade.html
 112 file path=usr/lib/help/auths/locale/LabelWinNoView.html
 113 file path=usr/lib/help/auths/locale/LabelWinUpgrade.html
 114 file path=usr/lib/help/auths/locale/LinkSecurity.html
 115 file path=usr/lib/help/auths/locale/LoginEnable.html
 116 file path=usr/lib/help/auths/locale/LoginHeader.html
 117 file path=usr/lib/help/auths/locale/LoginRemote.html
 118 file path=usr/lib/help/auths/locale/MailHeader.html
 119 file path=usr/lib/help/auths/locale/MailQueue.html
 120 file path=usr/lib/help/auths/locale/NetworkAutoconfRead.html
 121 file path=usr/lib/help/auths/locale/NetworkAutoconfSelect.html
 122 file path=usr/lib/help/auths/locale/NetworkAutoconfWlan.html
 123 file path=usr/lib/help/auths/locale/NetworkAutoconfWrite.html
 124 file path=usr/lib/help/auths/locale/NetworkHeader.html
 125 file path=usr/lib/help/auths/locale/NetworkILBconf.html
 126 file path=usr/lib/help/auths/locale/NetworkILBenable.html
```

```
127 file path=usr/lib/help/auths/locale/NetworkInterfaceConfig.html
128 file path=usr/lib/help/auths/locale/NetworkVRRP.html
129 file path=usr/lib/help/auths/locale/PriAdmin.html
130 file path=usr/lib/help/auths/locale/PrintAdmin.html
131 file path=usr/lib/help/auths/locale/PrintCancel.html
132 file path=usr/lib/help/auths/locale/PrintHeader.html
133 file path=usr/lib/help/auths/locale/PrintList.html
134 file path=usr/lib/help/auths/locale/PrintNoBanner.html
135 file path=usr/lib/help/auths/locale/PrintPs.html
136 file path=usr/lib/help/auths/locale/PrintUnlabeled.html
137 file path=usr/lib/help/auths/locale/ProfmgrHeader.html
138 file path=usr/lib/help/auths/locale/RoleHeader.html
139 file path=usr/lib/help/auths/locale/SmfAllocate.html
140 file path=usr/lib/help/auths/locale/SmfAutofsStates.html
141 file path=usr/lib/help/auths/locale/SmfCoreadmStates.html
142 file path=usr/lib/help/auths/locale/SmfCronStates.html
143 file path=usr/lib/help/auths/locale/SmfExAcctFlowStates.html
144 file path=usr/lib/help/auths/locale/SmfExAcctNetStates.html
145 file path=usr/lib/help/auths/locale/SmfExAcctProcessStates.html
146 file path=usr/lib/help/auths/locale/SmfExAcctTaskStates.html
147 file path=usr/lib/help/auths/locale/SmfHeader.html
148 file path=usr/lib/help/auths/locale/SmfILBStates.html
149 file path=usr/lib/help/auths/locale/SmfIPsecStates.html
150 file path=usr/lib/help/auths/locale/SmfIdmapStates.html
151 file path=usr/lib/help/auths/locale/SmfInetdStates.html
152 file path=usr/lib/help/auths/locale/SmfLocationStates.html
153 file path=usr/lib/help/auths/locale/SmfMDNSStates.html
154 file path=usr/lib/help/auths/locale/SmfManageAudit.html
155 file path=usr/lib/help/auths/locale/SmfManageHeader.html
156 file path=usr/lib/help/auths/locale/SmfManageHotplug.html
157 file path=usr/lib/help/auths/locale/SmfManageZFSSnap.html
158 file path=usr/lib/help/auths/locale/SmfModifyAppl.html
159 file path=usr/lib/help/auths/locale/SmfModifyDepend.html
160 file path=usr/lib/help/auths/locale/SmfModifyFramework.html
161 file path=usr/lib/help/auths/locale/SmfModifyHeader.html
162 file path=usr/lib/help/auths/locale/SmfModifyMethod.html
163 file path=usr/lib/help/auths/locale/SmfNADDStates.html
164 file path=usr/lib/help/auths/locale/SmfNDMPStates.html
165 file path=usr/lib/help/auths/locale/SmfNWAMStates.html
166 file path=usr/lib/help/auths/locale/SmfNscdStates.html
167 file path=usr/lib/help/auths/locale/SmfPowerStates.html
168 file path=usr/lib/help/auths/locale/SmfReparseStates.html
169 file path=usr/lib/help/auths/locale/SmfRoutingStates.html
170 file path=usr/lib/help/auths/locale/SmfSMBFSStates.html
171 file path=usr/lib/help/auths/locale/SmfSMBStates.html
172 file path=usr/lib/help/auths/locale/SmfSendmailStates.html
173 file path=usr/lib/help/auths/locale/SmfSshStates.html
174 file path=usr/lib/help/auths/locale/SmfSyslogStates.html
175 file path=usr/lib/help/auths/locale/SmfVRRPStates.html
176 file path=usr/lib/help/auths/locale/SmfValueAudit.html
177 file path=usr/lib/help/auths/locale/SmfValueCoreadm.html
178 file path=usr/lib/help/auths/locale/SmfValueExAcctFlow.html
179 file path=usr/lib/help/auths/locale/SmfValueExAcctNet.html
180 file path=usr/lib/help/auths/locale/SmfValueExAcctProcess.html
181 file path=usr/lib/help/auths/locale/SmfValueExAcctTask.html
182 file path=usr/lib/help/auths/locale/SmfValueFirewall.html
183 file path=usr/lib/help/auths/locale/SmfValueHeader.html
184 file path=usr/lib/help/auths/locale/SmfValueIPsec.html
185 file path=usr/lib/help/auths/locale/SmfValueIdmap.html
186 file path=usr/lib/help/auths/locale/SmfValueInetd.html
187 file path=usr/lib/help/auths/locale/SmfValueMDNS.html
188 file path=usr/lib/help/auths/locale/SmfValueNADD.html
189 file path=usr/lib/help/auths/locale/SmfValueNDMP.html
190 file path=usr/lib/help/auths/locale/SmfValueNWAM.html
191 file path=usr/lib/help/auths/locale/SmfValueRouting.html
192 file path=usr/lib/help/auths/locale/SmfValueSMB.html
```



```
193 file path=usr/lib/help/auths/locale/SmfValueVscan.html
194 file path=usr/lib/help/auths/locale/SmfValueVt.html
195 file path=usr/lib/help/auths/locale/SmfVscanStates.html
196 file path=usr/lib/help/auths/locale/SmfVtStates.html
197 file path=usr/lib/help/auths/locale/SmfWpaStates.html
198 file path=usr/lib/help/auths/locale/SysCpuPowerMgmt.html
199 file path=usr/lib/help/auths/locale/SysDate.html
200 file path=usr/lib/help/auths/locale/SysHeader.html
201 file path=usr/lib/help/auths/locale/SysMaintenance.html
202 file path=usr/lib/help/auths/locale/SysPowerMgmtBrightness.html
203 file path=usr/lib/help/auths/locale/SysPowerMgmtHeader.html
204 file path=usr/lib/help/auths/locale/SysPowerMgmtSuspend.html
205 file path=usr/lib/help/auths/locale/SysPowerMgmtSuspendtoDisk.html
206 file path=usr/lib/help/auths/locale/SysPowerMgmtSuspendtoRAM.html
207 file path=usr/lib/help/auths/locale/SysShutdown.html
208 file path=usr/lib/help/auths/locale/SysSyseventRead.html
209 file path=usr/lib/help/auths/locale/SysSyseventWrite.html
210 file path=usr/lib/help/auths/locale/TNDaemon.html
211 file path=usr/lib/help/auths/locale/TNctl.html
212 file path=usr/lib/help/auths/locale/ValueTND.html
213 file path=usr/lib/help/auths/locale/WifiConfig.html
214 file path=usr/lib/help/auths/locale/WifiWep.html
213 file path=usr/lib/help/auths/locale/ZoneCloneFrom.html
214 file path=usr/lib/help/auths/locale/ZoneHeader.html
215 file path=usr/lib/help/auths/locale/ZoneLogin.html
216 file path=usr/lib/help/auths/locale/ZoneManage.html
217 file path=usr/lib/help/profiles/locale/RtAcctadm.html
218 file path=usr/lib/help/profiles/locale/RtAll.html
219 file path=usr/lib/help/profiles/locale/RtAuditCfg.html
220 file path=usr/lib/help/profiles/locale/RtAuditCtrl.html
221 file path=usr/lib/help/profiles/locale/RtAuditReview.html
222 file path=usr/lib/help/profiles/locale/RtCPUPowerManagement.html
223 file path=usr/lib/help/profiles/locale/RtConsUser.html
224 file path=usr/lib/help/profiles/locale/RtContractObserver.html
225 file path=usr/lib/help/profiles/locale/RtCronMngmnt.html
226 file path=usr/lib/help/profiles/locale/RtCryptoMngmnt.html
227 file path=usr/lib/help/profiles/locale/RtDHCPMngmnt.html
228 file path=usr/lib/help/profiles/locale/RtDatAdmin.html
229 file path=usr/lib/help/profiles/locale/RtDefault.html
230 file path=usr/lib/help/profiles/locale/RtDeviceMngmnt.html
231 file path=usr/lib/help/profiles/locale/RtDeviceSecurity.html
232 file path=usr/lib/help/profiles/locale/RtExAcctFlow.html
233 file path=usr/lib/help/profiles/locale/RtExAcctNet.html
234 file path=usr/lib/help/profiles/locale/RtExAcctProcess.html
235 file path=usr/lib/help/profiles/locale/RtExAcctTask.html
236 file path=usr/lib/help/profiles/locale/RtFTPMngmnt.html
237 file path=usr/lib/help/profiles/locale/RtFileSysMngmnt.html
238 file path=usr/lib/help/profiles/locale/RtFileSysSecurity.html
239 file path=usr/lib/help/profiles/locale/RtHotplugMngmnt.html
240 file path=usr/lib/help/profiles/locale/RtIPFilterMngmnt.html
241 file path=usr/lib/help/profiles/locale/RtIdmapMngmnt.html
242 file path=usr/lib/help/profiles/locale/RtIdmapNameRulesMngmnt.html
243 file path=usr/lib/help/profiles/locale/RtInetdMngmnt.html
244 file path=usr/lib/help/profiles/locale/RtInfoSec.html
245 file path=usr/lib/help/profiles/locale/RtKerberosClntMngmnt.html
246 file path=usr/lib/help/profiles/locale/RtKerberosSrvrMngmnt.html
247 file path=usr/lib/help/profiles/locale/RtLogMngmnt.html
248 file path=usr/lib/help/profiles/locale/RtMailMngmnt.html
249 file path=usr/lib/help/profiles/locale/RtMaintAndRepair.html
250 file path=usr/lib/help/profiles/locale/RtMediaBkup.html
251 file path=usr/lib/help/profiles/locale/RtMediaCtlg.html
252 file path=usr/lib/help/profiles/locale/RtMediaRestore.html
253 file path=usr/lib/help/profiles/locale/RtNDMPMngmnt.html
254 file path=usr/lib/help/profiles/locale/RtNameServiceAdmin.html
255 file path=usr/lib/help/profiles/locale/RtNameServiceSecure.html
256 file path=usr/lib/help/profiles/locale/RtNetAutoconfAdmin.html
```

```
257 file path=usr/lib/help/profiles/locale/RtNetAutoconfUser.html
258 file path=usr/lib/help/profiles/locale/RtNetILB.html
259 file path=usr/lib/help/profiles/locale/RtNetIPsec.html
260 file path=usr/lib/help/profiles/locale/RtNetLinkSecure.html
261 file path=usr/lib/help/profiles/locale/RtNetMngmnt.html
262 file path=usr/lib/help/profiles/locale/RtNetObservability.html
263 file path=usr/lib/help/profiles/locale/RtNetSecure.html
264 file path=usr/lib/help/profiles/locale/RtNetVRRP.html
265 file path=usr/lib/help/profiles/locale/RtNetWifiMngmnt.html
266 file path=usr/lib/help/profiles/locale/RtNetWifiSecure.html
267 file path=usr/lib/help/profiles/locale/RtObAccessMngmnt.html
268 file path=usr/lib/help/profiles/locale/RtObjectLabelMngmnt.html
269 file path=usr/lib/help/profiles/locale/RtOperator.html
270 file path=usr/lib/help/profiles/locale/RtOutsideAccred.html
271 file path=usr/lib/help/profiles/locale/RtPriAdmin.html
272 file path=usr/lib/help/profiles/locale/RtPrntAdmin.html
273 file path=usr/lib/help/profiles/locale/RtProcManagement.html
274 file path=usr/lib/help/profiles/locale/RtReparseMngmnt.html
275 file path=usr/lib/help/profiles/locale/RtReservedProfile.html
276 file path=usr/lib/help/profiles/locale/RtRightsDelegate.html
277 file path=usr/lib/help/profiles/locale/RtSMBFSMngmnt.html
278 file path=usr/lib/help/profiles/locale/RtSMBMngmnt.html
279 file path=usr/lib/help/profiles/locale/RtSoftwareInstall.html
280 file path=usr/lib/help/profiles/locale/RtSysAdmin.html
281 file path=usr/lib/help/profiles/locale/RtSysEvMngmnt.html
282 file path=usr/lib/help/profiles/locale/RtSysPowerMgmt.html
283 file path=usr/lib/help/profiles/locale/RtSysPowerMgmtBrightness.html
284 file path=usr/lib/help/profiles/locale/RtSysPowerMgmtSuspend.html
285 file path=usr/lib/help/profiles/locale/RtSysPowerMgmtSuspendtoDisk.html
286 file path=usr/lib/help/profiles/locale/RtSysPowerMgmtSuspendtoRAM.html
287 file path=usr/lib/help/profiles/locale/RtUserMngmnt.html
288 file path=usr/lib/help/profiles/locale/RtUserSecurity.html
289 file path=usr/lib/help/profiles/locale/RtVscanMngmnt.html
290 file path=usr/lib/help/profiles/locale/RtZFSFileSysMngmnt.html
291 file path=usr/lib/help/profiles/locale/RtZFSStorageMngmnt.html
292 file path=usr/lib/help/profiles/locale/RtZoneMngmnt.html
293 file path=usr/lib/help/profiles/locale/RtZoneSecurity.html
294 file path=usr/lib/locale/C/LC_MESSAGES/AMD.po
295 file path=usr/lib/locale/C/LC_MESSAGES/DISK.po
296 file path=usr/lib/locale/C/LC_MESSAGES/FMD.po
297 file path=usr/lib/locale/C/LC_MESSAGES/FMNOTIFY.po
298 file path=usr/lib/locale/C/LC_MESSAGES/GMCA.po
299 file path=usr/lib/locale/C/LC_MESSAGES/INTEL.po
300 file path=usr/lib/locale/C/LC_MESSAGES/NXGE.po
301 file path=usr/lib/locale/C/LC_MESSAGES/PCI.po
302 file path=usr/lib/locale/C/LC_MESSAGES/PCIEX.po
303 file path=usr/lib/locale/C/LC_MESSAGES/SCA1000.po
304 file path=usr/lib/locale/C/LC_MESSAGES/SCA500.po
305 file path=usr/lib/locale/C/LC_MESSAGES/SCF.po
306 file path=usr/lib/locale/C/LC_MESSAGES/SENSOR.po
307 file path=usr/lib/locale/C/LC_MESSAGES/SMF.po
308 file path=usr/lib/locale/C/LC_MESSAGES/STORAGE.po
309 file path=usr/lib/locale/C/LC_MESSAGES/SUN4.po
310 file path=usr/lib/locale/C/LC_MESSAGES/SUN4U.po
311 file path=usr/lib/locale/C/LC_MESSAGES/SUN4V.po
312 file path=usr/lib/locale/C/LC_MESSAGES/SUNOS.po
313 file path=usr/lib/locale/C/LC_MESSAGES/SUNW_OST_ADMIN.po group=sys
314 file path=usr/lib/locale/C/LC_MESSAGES/SUNW_OST_LINFO group=sys
315 file path=usr/lib/locale/C/LC_MESSAGES/SUNW_OST_NETRPC.po group=sys
316 file path=usr/lib/locale/C/LC_MESSAGES/SUNW_OST_OSCMD.po group=sys
317 file path=usr/lib/locale/C/LC_MESSAGES/SUNW_OST_OSLIB.po group=sys
318 file path=usr/lib/locale/C/LC_MESSAGES/SUNW_OST_SGS.po group=sys
319 file path=usr/lib/locale/C/LC_MESSAGES/SUNW_OST_SYSOSPAM.po group=sys
320 file path=usr/lib/locale/C/LC_MESSAGES/SUNW_OST_UCBCMD.po group=sys
321 file path=usr/lib/locale/C/LC_MESSAGES/SUNW_OST_ZONEINFO.po group=sys
322 file path=usr/lib/locale/C/LC_MESSAGES/ZFS.po
```



```
323 file path=usr/lib/locale/C/LC_MESSAGES/libast group=sys
324 file path=usr/lib/locale/C/LC_MESSAGES/libcmd group=sys
325 file path=usr/lib/locale/C/LC_MESSAGES/libdll group=sys
326 file path=usr/lib/locale/C/LC_MESSAGES/libshell group=sys
327 file path=usr/lib/locale/C/LC_MESSAGES/libsum group=sys
328 file path=usr/lib/locale/C/LC_MESSAGES/magic group=sys
329 file path=usr/lib/locale/C/LC_MESSAGES/mailx.help group=sys
330 file path=usr/lib/locale/C/LC_MESSAGES/more.help group=sys
331 file path=usr/lib/locale/C/LC_MESSAGES/priv_names group=sys
332 file path=usr/lib/locale/C/LC_MESSAGES/uxlibc.src group=sys
333 file path=usr/lib/locale/C/LC_TIME/SUNW_OST_OSCMD.po group=sys
334 file path=usr/lib/locale/C/LC_TIME/SUNW_OST_OSLIB.po group=sys
335 file path=usr/share/lib/locale/com/sun/admin/pm/client/pmHelpResources.java \
336     group=lp
337 file path=usr/share/lib/locale/com/sun/admin/pm/client/pmResources.java \
338     group=lp
339 file \
340     path=usr/share/lib/locale/com/sun/dhcpmgr/bridge/ResourceBundle.properties
341 file \
342     path=usr/share/lib/locale/com/sun/dhcpmgr/cli/common/ResourceBundle.properti
343 file \
344     path=usr/share/lib/locale/com/sun/dhcpmgr/cli/dhcpbatch/ResourceBundle.prope
345 file \
346     path=usr/share/lib/locale/com/sun/dhcpmgr/cli/dhcpconfig/ResourceBundle.prop
347 file \
348     path=usr/share/lib/locale/com/sun/dhcpmgr/cli/dhtadm/ResourceBundle.properti
349 file \
350     path=usr/share/lib/locale/com/sun/dhcpmgr/cli/pntadm/ResourceBundle.properti
351 file \
352     path=usr/share/lib/locale/com/sun/dhcpmgr/client/OptionDescriptions.properti
353 file \
354     path=usr/share/lib/locale/com/sun/dhcpmgr/client/ResourceBundle.properties
355 file \
356     path=usr/share/lib/locale/com/sun/dhcpmgr/client/SUNWbinfiles/ResourceBundle
357 file \
358     path=usr/share/lib/locale/com/sun/dhcpmgr/client/SUNWfiles/ResourceBundle.pr
359 file path=usr/share/lib/locale/com/sun/dhcpmgr/client/help/art/bannersmc.gif
360 file path=usr/share/lib/locale/com/sun/dhcpmgr/client/help/art/dot1.gif
361 file path=usr/share/lib/locale/com/sun/dhcpmgr/client/help/art/dot2.gif
362 file path=usr/share/lib/locale/com/sun/dhcpmgr/client/help/art/folder.gif
363 file path=usr/share/lib/locale/com/sun/dhcpmgr/client/help/art/macro2.gif
364 file path=usr/share/lib/locale/com/sun/dhcpmgr/client/help/art/macroflow.gif
365 file path=usr/share/lib/locale/com/sun/dhcpmgr/client/help/art/tip2.gif
366 file \
367     path=usr/share/lib/locale/com/sun/dhcpmgr/client/help/dhcp_addr_create.html
368 file path=usr/share/lib/locale/com/sun/dhcpmgr/client/help/dhcp_addr_del.html
369 file path=usr/share/lib/locale/com/sun/dhcpmgr/client/help/dhcp_addr_dup.html
370 file path=usr/share/lib/locale/com/sun/dhcpmgr/client/help/dhcp_addr_how.html
371 file path=usr/share/lib/locale/com/sun/dhcpmgr/client/help/dhcp_addr_mod.html
372 file \
373     path=usr/share/lib/locale/com/sun/dhcpmgr/client/help/dhcp_addr_multi.html
374 file path=usr/share/lib/locale/com/sun/dhcpmgr/client/help/dhcp_addr_ref.html
375 file path=usr/share/lib/locale/com/sun/dhcpmgr/client/help/dhcp_addr_rel.html
376 file path=usr/share/lib/locale/com/sun/dhcpmgr/client/help/dhcp_addr_view.html
377 file path=usr/share/lib/locale/com/sun/dhcpmgr/client/help/dhcp_addr_wiz.html
378 file \
379     path=usr/share/lib/locale/com/sun/dhcpmgr/client/help/dhcp_config_wiz.html
380 file \
381     path=usr/share/lib/locale/com/sun/dhcpmgr/client/help/dhcp_convert_wiz.html
382 file \
383     path=usr/share/lib/locale/com/sun/dhcpmgr/client/help/dhcp_export_wiz.html
384 file \
385     path=usr/share/lib/locale/com/sun/dhcpmgr/client/help/dhcp_import_wiz.html
386 file \
387     path=usr/share/lib/locale/com/sun/dhcpmgr/client/help/dhcp_macro_create.html
388 file path=usr/share/lib/locale/com/sun/dhcpmgr/client/help/dhcp_macro_del.html
```

```
389 file path=usr/share/lib/locale/com/sun/dhcpmgr/client/help/dhcp_macro_dup.html
390 file path=usr/share/lib/locale/com/sun/dhcpmgr/client/help/dhcp_macro_how.html
391 file path=usr/share/lib/locale/com/sun/dhcpmgr/client/help/dhcp_macro_mod.html
392 file path=usr/share/lib/locale/com/sun/dhcpmgr/client/help/dhcp_macro_ref.html
393 file \
394     path=usr/share/lib/locale/com/sun/dhcpmgr/client/help/dhcp_macro_view.html
395 file \
396     path=usr/share/lib/locale/com/sun/dhcpmgr/client/help/dhcp_macros_about.html
397 file path=usr/share/lib/locale/com/sun/dhcpmgr/client/help/dhcp_main_hlp.html
398 file path=usr/share/lib/locale/com/sun/dhcpmgr/client/help/dhcp_main_how.html
399 file path=usr/share/lib/locale/com/sun/dhcpmgr/client/help/dhcp_main_idx.html
400 file \
401     path=usr/share/lib/locale/com/sun/dhcpmgr/client/help/dhcp_main_menus.html
402 file path=usr/share/lib/locale/com/sun/dhcpmgr/client/help/dhcp_main_top.html
403 file path=usr/share/lib/locale/com/sun/dhcpmgr/client/help/dhcp_net_del.html
404 file path=usr/share/lib/locale/com/sun/dhcpmgr/client/help/dhcp_net_ref.html
405 file path=usr/share/lib/locale/com/sun/dhcpmgr/client/help/dhcp_net_wiz.html
406 file \
407     path=usr/share/lib/locale/com/sun/dhcpmgr/client/help/dhcp_option_create.htm
408 file \
409     path=usr/share/lib/locale/com/sun/dhcpmgr/client/help/dhcp_option_del.html
410 file \
411     path=usr/share/lib/locale/com/sun/dhcpmgr/client/help/dhcp_option_dup.html
412 file \
413     path=usr/share/lib/locale/com/sun/dhcpmgr/client/help/dhcp_option_how.html
414 file \
415     path=usr/share/lib/locale/com/sun/dhcpmgr/client/help/dhcp_option_mod.html
416 file \
417     path=usr/share/lib/locale/com/sun/dhcpmgr/client/help/dhcp_option_ref.html
418 file \
419     path=usr/share/lib/locale/com/sun/dhcpmgr/client/help/dhcp_option_tags.html
420 file \
421     path=usr/share/lib/locale/com/sun/dhcpmgr/client/help/dhcp_option_view.html
422 file \
423     path=usr/share/lib/locale/com/sun/dhcpmgr/client/help/dhcp_relay_choose.html
424 file \
425     path=usr/share/lib/locale/com/sun/dhcpmgr/client/help/dhcp_relay_config.html
426 file path=usr/share/lib/locale/com/sun/dhcpmgr/client/help/dhcp_relay_dis.html
427 file \
428     path=usr/share/lib/locale/com/sun/dhcpmgr/client/help/dhcp_relay_enable.html
429 file path=usr/share/lib/locale/com/sun/dhcpmgr/client/help/dhcp_relay_how.html
430 file path=usr/share/lib/locale/com/sun/dhcpmgr/client/help/dhcp_relay_ref.html
431 file \
432     path=usr/share/lib/locale/com/sun/dhcpmgr/client/help/dhcp_relay_serv.html
433 file \
434     path=usr/share/lib/locale/com/sun/dhcpmgr/client/help/dhcp_relay_unconfig.ht
435 file \
436     path=usr/share/lib/locale/com/sun/dhcpmgr/client/help/dhcp_server_serv.html
437 file \
438     path=usr/share/lib/locale/com/sun/dhcpmgr/client/help/dhcp_server_unconfig.h
439 file \
440     path=usr/share/lib/locale/com/sun/dhcpmgr/client/help/dhcp_solaris_about.htm
441 file \
442     path=usr/share/lib/locale/com/sun/dhcpmgr/common/ResourceBundle.properties
443 file path=usr/share/lib/locale/com/sun/dhcpmgr/data/ResourceBundle.properties
444 file path=usr/share/lib/locale/com/sun/dhcpmgr/ui/ResourceBundle.properties
445 file path=usr/share/lib/locale/com/sun/slp/ClientLib_en.properties group=sys
446 file path=usr/share/lib/locale/com/sun/slp/Server_en.properties group=sys
447 legacy pkg=SUNW0on arch=all \
448     desc="localizable message files for the OS-Networking consolidation" \
449     name="Localizable ON message files" version=11.11,REV=2009.11.10
450 license cr_Sun license=cr_Sun
451 license lic_CDDL license=lic_CDDL
```

```
**********************************************************
    2299 Tue Jun 12 19:55:14 2012
new/usr/src/pkg/manifests/service-network-wpa_supplicant.mf
wpa_supplicant pkg now is created correctly in illumos-gate
wpad renamed to wpa_supplicant
**********************************************************
```

```
#
# 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) 2010, Oracle and/or its affiliates. All rights reserved.
#

set name=pkg.fmri value=pkg:/service/network/wpa_supplicant@$(PKGVERS)
set name=pkg.description \
    value="The service implements the IEEE802.11i (WPA/WPA2) specification."
set name=pkg.summary value="Wireless WPA Supplicant"
set name=info.classification \
    value=org.opensolaris.category.2008:System/Hardware
set name=variant.arch value=$(ARCH)
dir path=lib
dir path=lib/svc
dir path=lib/svc/manifest group=sys
dir path=lib/svc/manifest/network group=sys
dir path=usr group=sys
dir path=usr/lib
dir path=usr/lib/inet
dir path=usr/share/man/man1m
file path=lib/svc/manifest/network/wpa_supplicant.xml group=sys mode=0444
file path=usr/lib/inet/wpa_supplicant mode=0555
file path=usr/share/man/man1m/wpa_supplicant.1m
legacy pkg=SUNWwpar \
    desc="The service implements the IEEE802.11i (WPA/WPA2) specification." \
    name="Wireless WPA Supplicant, (Root)"
legacy pkg=SUNWwpau \
    desc="The service implements the IEEE802.11i (WPA/WPA2) specification." \
    name="Wireless WPA Supplicant, (Usr)"
license cr_Sun license=cr_Sun
license lic_CDDL license=lic_CDDL
license usr/src/cmd/cmd-inet/usr.lib/wpa_supplicant/THIRDPARTYLICENSE \
    license=usr/src/cmd/cmd-inet/usr.lib/wpa_supplicant/THIRDPARTYLICENSE
depend fmri=SUNWcs type=require
depend fmri=SUNWcsd type=require
depend fmri=system/library type=require
depend fmri=system/library/platform type=require
depend fmri=system/library/processor type=require
#endif /* ! codereview */
```

```
**********************************************************
   12294 Tue Jun 12 19:55:15 2012
new/usr/src/uts/common/inet/wifi_ioctl.h
ess_list ioctl now provides all scan results properties for wpa/libdlwlan
first integration of wpa_s control interface client code
first integration of wpa_s wpa_ie parsing code
**********************************************************
```

```
/*
 * 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.
 */

/*
 * Macro and date structures defined for 802.11 wifi config tool.
 * (no more used, we should remove them)
#endif /* ! codereview */
 */

#ifndef __WIFI_IOCTL_H
#define __WIFI_IOCTL_H

#include <sys/types.h>

#ifdef  __cplusplus
extern "C" {
#endif

#define MAX_KEY_LENGTH 26
#define MAX_ESSID_LENGTH (32 + 1)       /* max essid length is 32 */
                                        /* one more for '\0' */
#define MAX_CHANNEL_NUM 99
#define MAX_RSSI 15
#define MAX_NWEPKEYS 4
#define NET_802_11 80211
#define MAX_BUF_LEN 65536
#define MAX_SCAN_SUPPORT_RATES 8

/*
 * ioctls
 */

#endif /* ! codereview */
#define WLAN_IOCTL_BASE 0x1000
#define WLAN_GET_VERSION (WLAN_IOCTL_BASE + 0x0)
#define WLAN_SET_PARAM (WLAN_IOCTL_BASE + 0x2)
#define WLAN_GET_PARAM (WLAN_IOCTL_BASE + 0x3)
```



```
#define WLAN_COMMAND (WLAN_IOCTL_BASE + 0x4)

/*
 * parameters
 */

#endif /* ! codereview */
#define WL_PARAMETERS_BASE 0x2000
#define WL_BSSID (WL_PARAMETERS_BASE + 0x0)
#define WL_ESSID (WL_PARAMETERS_BASE + 0x1)
#define WL_NODE_NAME (WL_PARAMETERS_BASE + 0x2)
#define WL_PHY_SUPPORT (WL_PARAMETERS_BASE + 0x3)
#define WL_PHY_CONFIG (WL_PARAMETERS_BASE + 0x4)
#define WL_DOMAIN (WL_PARAMETERS_BASE + 0x5)
#define WL_POWER_MODE (WL_PARAMETERS_BASE + 0x6)
#define WL_TX_POWER (WL_PARAMETERS_BASE + 0x7)
#define WL_RSSI (WL_PARAMETERS_BASE + 0x8)
#define WL_RSSI_THRESHOLD (WL_PARAMETERS_BASE + 0x9)
#define WL_ESS_LIST (WL_PARAMETERS_BASE + 0xa)
#define WL_BSS_TYPE (WL_PARAMETERS_BASE + 0xb)
#define WL_CREATE_IBSS (WL_PARAMETERS_BASE + 0xc)
#define WL_RTS_THRESHOLD (WL_PARAMETERS_BASE + 0xd)
#define WL_SHORT_RETRY (WL_PARAMETERS_BASE + 0xe)
#define WL_LONG_RETRY (WL_PARAMETERS_BASE + 0xf)
#define WL_BEACON_PERIOD (WL_PARAMETERS_BASE + 0x10)
#define WL_TX_LIFETIME (WL_PARAMETERS_BASE + 0x11)
#define WL_RX_LIFETIME (WL_PARAMETERS_BASE + 0x12)
#define WL_FRAG_THRESHOLD (WL_PARAMETERS_BASE + 0x13)
#define WL_VENDOR_ID (WL_PARAMETERS_BASE + 0x14)
#define WL_PRODUCT_ID (WL_PARAMETERS_BASE + 0x15)
#define WL_NUM_ANTS (WL_PARAMETERS_BASE + 0x16)
#define WL_RX_ANTENNA (WL_PARAMETERS_BASE + 0x17)
#define WL_TX_ANTENNA (WL_PARAMETERS_BASE + 0x18)
#define WL_SUPPORTED_RATES (WL_PARAMETERS_BASE + 0x19)
#define WL_DESIRED_RATES (WL_PARAMETERS_BASE + 0x1a)
#define WL_WEP_KEY_TAB (WL_PARAMETERS_BASE + 0x1b)
#define WL_WEP_KEY_ID (WL_PARAMETERS_BASE + 0x1c)
#define WL_WEP_MAPPING_TAB (WL_PARAMETERS_BASE + 0x1d)
#define WL_WEP_MAPPING_LEN (WL_PARAMETERS_BASE + 0x1e)
#define WL_ENCRYPTION (WL_PARAMETERS_BASE + 0x1f)
#define WL_AUTH_MODE (WL_PARAMETERS_BASE + 0x20)
#define WL_EXCL_UNENC (WL_PARAMETERS_BASE + 0x21)
#define WL_RFMON (WL_PARAMETERS_BASE + 0x22)
#define WL_RADIO (WL_PARAMETERS_BASE + 0x23)
#define WL_LINKSTATUS (WL_PARAMETERS_BASE + 0x24)
#define WL_DEV_DEPEND (WL_PARAMETERS_BASE + 0x25)

#endif /* ! codereview */
/*
 * commands
 */
#define WL_COMMAND_BASE 0x3000
#define WL_SCAN (WL_COMMAND_BASE + 0x0)
#define WL_DISASSOCIATE (WL_COMMAND_BASE + 0x1)
#define WL_REASSOCIATE (WL_COMMAND_BASE + 0x2)
#define WL_LOAD_DEFAULTS (WL_COMMAND_BASE + 0x3)
#define WL_ASSOCIAT (WL_COMMAND_BASE + 0x4)

/*
 * domains
 * outdated?
#endif /* ! codereview */
 */
/* --USA */
#define WL_DOMAIN_BASE 0x4000
#define WL_DOMAIN_FCC (WL_DOMAIN_BASE + 0x0)
```

```
/* --Canada */
#define WL_DOMAIN_DOC (WL_DOMAIN_BASE + 0x1)
/* --Most of Europe */
#define WL_DOMAIN_ETSI (WL_DOMAIN_BASE + 0x2)
/* --Spain */
#define WL_DOMAIN_SPAIN (WL_DOMAIN_BASE + 0x3)
/* --France */
#define WL_DOMAIN_FRANCE (WL_DOMAIN_BASE + 0x4)
/* --Japan */
#define WL_DOMAIN_MKK (WL_DOMAIN_BASE + 0x5)

/*
 * power mode
 * not implemented
#endif /* ! codereview */
 */

#define WL_PM_AM 0x0
#define WL_PM_MPS 0x1
#define WL_PM_FAST 0x2
#define WL_PM_USER 0x3

/*
 * rates
 */
#define WL_RATE_BASIC_SET 0x80
#define WL_RATE_1M 2
#define WL_RATE_2M 4
#define WL_RATE_5_5M 11
#define WL_RATE_6M 12
#define WL_RATE_9M 18
#define WL_RATE_11M 22
#define WL_RATE_12M 24
#define WL_RATE_18M 36
#define WL_RATE_22M 44
#define WL_RATE_24M 48
#define WL_RATE_33M 66
#define WL_RATE_36M 72
#define WL_RATE_48M 96
#define WL_RATE_54M 108
/*
 * wep operations
 */
#define WL_WEP_OPERATION_BASE 0x6000
#define WL_ADD (WL_WEP_OPERATION_BASE + 0x0)
#define WL_DEL (WL_WEP_OPERATION_BASE + 0x1)
#define WL_NUL (WL_WEP_OPERATION_BASE + 0x2)
#define WL_IND (WL_WEP_OPERATION_BASE + 0x3)

#define WL_NOENCRYPTION 0x0
#define WL_ENC_WEP 0x1
#define WL_ENC_WPA 0x2

#endif /* ! codereview */
#define WL_OPENSYSTEM 0x1
#define WL_SHAREDKEY 0x2

/*
 * linkstatus
 */
#define WL_CONNECTED 0x0
#define WL_NOTCONNECTED 0x1

/*
 * prives
 */
```



```
#define WL_PRIV_BASE 0x7000
#define WL_PRIV_RW (WL_PRIV_BASE + 0x0)
#define WL_PRIV_R (WL_PRIV_BASE + 0x1)
#define WL_PRIV_W (WL_PRIV_BASE + 0x2)
#define WL_PRIV_INT (WL_PRIV_BASE + 0x3)
#define WL_PRIV_INT_ARRAY (WL_PRIV_BASE + 0x4)
#define WL_PRIV_BYTE (WL_PRIV_BASE + 0x5)
#define WL_PRIV_BYTE_ARRAY (WL_PRIV_BASE + 0x6)
#define WL_PRIV_STRING (WL_PRIV_BASE + 0x7)
#define WL_PRIV_STRING_ARRAY (WL_PRIV_BASE + 0x8)
/*
 * return values
 */
#define WL_SUCCESS 0x0
#define WL_NOTSUPPORTED EINVAL
#define WL_LACK_FEATURE ENOTSUP
#define WL_HW_ERROR EIO
#define WL_ACCESS_DENIED EACCES
#define WL_RETURN_BASE  0x7000
#define WL_READONLY (WL_RETURN_BASE + 0x1)
#define WL_WRITEONLY (WL_RETURN_BASE + 0x2)
#define WL_NOAP (WL_RETURN_BASE + 0x3)
/*
 * other values
 */
#define WL_OTHER_BASE 0x8000
#define WL_FHSS (WL_OTHER_BASE + 0x0)
#define WL_DSSS (WL_OTHER_BASE + 0x1)
#define WL_IRBASE (WL_OTHER_BASE + 0x2)
#define WL_OFDM (WL_OTHER_BASE + 0x3)
#define WL_HRDS (WL_OTHER_BASE + 0x4)
#define WL_ERP (WL_OTHER_BASE + 0x5)

/* aligned with wpa_s values */
#define WL_BSS_BSS 0
#define WL_BSS_IBSS 1
#define WL_BSS_AP 2
#define WL_BSS_BSS 1
#define WL_BSS_IBSS 3
#define WL_BSS_ANY 2
/*
 * field_offset
 */
#define WIFI_BUF_OFFSET         offsetof(wldp_t, wldp_buf)

/*
 * type definationes
 */
typedef boolean_t wl_create_ibss_t;
typedef uint8_t wl_bssid_t[6];
typedef char wl_bssid_t[6];

typedef struct wl_essid {
        uint32_t wl_essid_length;
        uint8_t wl_essid_essid[32];
} wl_essid_t;
        char wl_essid_essid[34];
}wl_essid_t;

/* there are no consumers for wl_nodename prop */
#endif /* ! codereview */
typedef struct wl_nodename {
        uint32_t wl_nodename_length;
        char wl_nodename_name[32];
        char wl_nodename_name[34];
} wl_nodename_t;
```

*unchanged_portion_omitted_*

```
315 typedef uint32_t wl_linkstatus_t;
316 typedef uint32_t wl_tx_pwer_t;
317 typedef uint32_t wl_rssi_t;
318 typedef uint32_t wl_rssi_threshold_t;
319 typedef uint32_t wl_bss_type_t;
320 typedef uint8_t wl_authmode_t;
132 typedef uint32_t wl_authmode_t;
321 typedef uint32_t wl_encryption_t;
322 typedef uint32_t wl_wep_key_id_t;
323 typedef boolean_t wl_radio_t;
324 typedef uint32_t wl_rts_threshold_t;
325 typedef uint32_t wl_short_retry_t;
326 typedef uint32_t wl_long_retry_t;
327 typedef uint16_t wl_beacon_period_t;
328 typedef uint32_t wl_beacon_age_t;
329 typedef uint64_t wl_beacon_tsf_t;
139 typedef uint32_t wl_beacon_period_t;
330 typedef uint32_t wl_tx_lifetime_t;
331 typedef uint32_t wl_rx_lifetime_t;
332 typedef uint32_t wl_frag_threshold_t;
333 typedef char wl_vendor_t[128];
334 typedef char wl_product_t[128];
335 typedef uint32_t wl_num_ants_t;
336 typedef uint32_t wl_rx_antenna_t;
337 typedef uint32_t wl_tx_antenna_t;

339 typedef struct wl_rates {
340         uint32_t wl_rates_num;
341         char wl_rates_rates[1];
342 } wl_rates_t;

344 /*
345  * Macro and data structures defined for 802.11i.
346  */

348 typedef struct wl_wpa_ie {
349         uint32_t        wpa_ie_len;
350         uint8_t         wpa_ie[1];      /* it's the head of wpa_ie */
351 } wl_wpa_ie_t;

353 typedef struct wl_wpa {
354         uint32_t        wpa_flag;
355 } wl_wpa_t;

357 typedef struct wl_capability {
358         uint32_t        caps;
359 } wl_capability_t;

361 typedef uint16_t wl_ess_caps;

363 /*
364  * WPA/RSN get/set key request.
365  * ik_type  : wep/tkip/aes
366  * ik_keyix : should be between 0 and 3, 0 will be used as default key.
367  * ik_keylen: key length in bytes.
368  * ik_keydata and ik_keylen include the DATA key and MIC key.
369  * ik_keyrsc/ik_keytsc: rx/tx seq number.
370  */
371 #pragma pack(1)
372 typedef struct wl_key {
373         uint8_t         ik_type;
374         uint8_t         ik_pad;

376         uint16_t        ik_keyix;
377         uint8_t         ik_keylen;
```



```
378         uint8_t         ik_flags;

380         uint8_t         ik_macaddr[6];
381         uint64_t        ik_keyrsc;
382         uint64_t        ik_keytsc;

384         /* [IEEE80211_KEYBUF_SIZE+IEEE80211_MICBUF_SIZE] */
385         uint8_t         ik_keydata[32];
386 } wl_key_t;
387 #pragma pack()

389 typedef struct wl_del_key {
390         uint8_t         idk_keyix;
391         uint8_t         idk_macaddr[6];
392 } wl_del_key_t;

394 /*
395  * structure for WL_MLME state manipulation request.
396  * im_op: operations include auth/deauth/assoc/disassoc,
397  * im_reason: 802.11 reason code
398  */
399 typedef struct wl_mlme {
400         uint8_t         im_op;
401         uint16_t        im_reason;
402         uint8_t         im_macaddr[6];
403 } wl_mlme_t;

405 /*
406  * beacon, probe response
407  */

409 #pragma pack(1)
410 #endif /* ! codereview */
411 typedef struct wl_ess_conf {
412         uint32_t                wl_ess_conf_length;
413         wl_essid_t              wl_ess_conf_essid;
414         wl_bssid_t              wl_ess_conf_bssid;
415         wl_beacon_period_t      wl_ess_conf_beacon_period;
416         wl_beacon_tsf_t         wl_ess_conf_beacon_tsf;
417         wl_beacon_age_t         wl_ess_conf_beacon_age;
154         char wl_ess_conf_reserved[2];
155         wl_bss_type_t wl_ess_conf_bsstype;
156         wl_authmode_t wl_ess_conf_authmode;
157         boolean_t wl_ess_conf_wepenabled;
418         wl_rssi_t               wl_ess_conf_sl;
419         char                    wl_supported_rates[MAX_SCAN_SUPPORT_RATES];
420 #endif /* ! codereview */
421         union {
422                 wl_fhss_t       wl_phy_fhss_conf;
423                 wl_dsss_t       wl_phy_dsss_conf;
424                 wl_ofdm_t       wl_phy_ofdm_conf;
425                 wl_erp_t        wl_phy_erp_conf;
426         } wl_phy_conf;
427         /* ieee80211_node capinfo != ieee80211com caps */
428         wl_ess_caps             wl_ess_conf_caps;
429         uint32_t                wl_ess_conf_wpa_ie_len;
430         uint8_t                 wl_ess_conf_wpa_ie[40];
159         char wl_supported_rates[MAX_SCAN_SUPPORT_RATES];
431 } wl_ess_conf_t;
432 #pragma pack()
433 #endif /* ! codereview */

435 typedef struct wl_ess_list {
436         uint32_t wl_ess_list_num;
437         wl_ess_conf_t wl_ess_list_ess[1];
438 } wl_ess_list_t;
```

```
440 typedef struct wl_wep_key {
441         uint32_t wl_wep_length;
442         char wl_wep_key[MAX_KEY_LENGTH];
443         uint32_t wl_wep_operation;
444 } wl_wep_key_t;
445 typedef wl_wep_key_t wl_wep_key_tab_t[MAX_NWEPKEYS];

447 typedef struct wep_mapping {
448         uint32_t wl_wep_map_index;
449         boolean_t wl_wep_map_wepon;
450         char wl_wep_map_mac_addr[6];
451         char wl_wep_map_reserved[2];
452         wl_wep_key_t wl_wep_map_wepkey;
453 } wep_mapping_t;
454 typedef wep_mapping_t wep_mapping_tab_t[1];

456 typedef struct wl_priv_param {
457         char wl_priv_name[8];
458         uint32_t wl_priv_type;
459         uint32_t wl_priv_size;
460         char wl_priv_value[1];
461 } wl_priv_param_t;

463 typedef struct wl_dev_depend {
464         uint32_t wl_dev_depend_num;
465         uint32_t wl_dev_depend_ret_idx;
466         wl_priv_param_t wl_dev_depend_priv[1];
467 } wl_dev_depend_t;

469 typedef struct wlan_ver {
470         uint32_t wl_ver_major;
471         uint32_t wl_ver_minor;
472 } wlan_ver_t;

474 typedef struct wldp {
475         uint32_t wldp_length;
476         uint32_t wldp_type;
477         uint32_t wldp_result;
478         uint32_t wldp_id;
479         uint32_t wldp_buf[1];
480 } wldp_t;

482 #ifdef __cplusplus
483 }
484 #endif

486 #endif /* __WIFI_IOCTL_H */
```

****
37539 Tue Jun 12 19:55:16 2012
new/usr/src/uts/common/io/dld/dld_drv.c
secobjs types now are "wep, psk, eap, pin"
dladm_wlan_secmode_t and dladm_secobj_class_t are not related anymore
****
_____***unchanged_portion_omitted_***

```
1160 /* ARGSUSED */
1161 static int
1162 drv_ioc_secobj_set(void *karg, intptr_t arg, int mode, cred_t *cred, int *rvalp)
1163 {
1164         dld_ioc_secobj_set_t    *ssp = karg;
1165         dld_secobj_t            *sobjp, *objp;
1166         int                     err;

1168         sobjp = &ssp->ss_obj;

1170         if (sobjp->so_class == DLD_SECOBJ_CLASS_TLS)
1170         if (sobjp->so_class != DLD_SECOBJ_CLASS_WEP &&
1171             sobjp->so_class != DLD_SECOBJ_CLASS_WPA)
1171                 return (EINVAL);

1173         if (sobjp->so_name[DLD_SECOBJ_NAME_MAX - 1] != '\0' ||
1174             sobjp->so_len > DLD_SECOBJ_VAL_MAX)
1175                 return (EINVAL);

1177         rw_enter(&drv_secobj_lock, RW_WRITER);
1178         err = mod_hash_find(drv_secobj_hash, (mod_hash_key_t)sobjp->so_name,
1179             (mod_hash_val_t *)&objp);
1180         if (err == 0) {
1181                 if ((ssp->ss_flags & DLD_SECOBJ_OPT_CREATE) != 0) {
1182                         rw_exit(&drv_secobj_lock);
1183                         return (EEXIST);
1184                 }
1185         } else {
1186                 ASSERT(err == MH_ERR_NOTFOUND);
1187                 if ((ssp->ss_flags & DLD_SECOBJ_OPT_CREATE) == 0) {
1188                         rw_exit(&drv_secobj_lock);
1189                         return (ENOENT);
1190                 }
1191                 objp = kmem_cache_alloc(drv_secobj_cachep, KM_SLEEP);
1192                 (void) strlcpy(objp->so_name, sobjp->so_name,
1193                     DLD_SECOBJ_NAME_MAX);

1195                 VERIFY(mod_hash_insert(drv_secobj_hash,
1196                     (mod_hash_key_t)objp->so_name, (mod_hash_val_t)objp) == 0);
1197         }
1198         bcopy(sobjp->so_val, objp->so_val, sobjp->so_len);
1199         objp->so_len = sobjp->so_len;
1200         objp->so_class = sobjp->so_class;
1201         rw_exit(&drv_secobj_lock);
1202         return (0);
1203 }
```

_____***unchanged_portion_omitted_***

```
**********************************************************
  212772 Tue Jun 12 19:55:18 2012
new/usr/src/uts/common/io/mac/mac.c
ess_list ioctl now provides all scan results properties for wpa/libdlwlan
first integration of wpa_s control interface client code
first integration of wpa_s wpa_ie parsing code
**********************************************************
/*
 * 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.
 */

/*
 * MAC Services Module
 *
 * The GLDv3 framework locking -  The MAC layer
 * --------------------------------------------
 *
 * The MAC layer is central to the GLD framework and can provide the locking
 * framework needed for itself and for the use of MAC clients. MAC end points
 * are fairly disjoint and don't share a lot of state. So a coarse grained
 * multi-threading scheme is to single thread all create/modify/delete or set
 * type of control operations on a per mac end point while allowing data threads
 * concurrently.
 *
 * Control operations (set) that modify a mac end point are always serialized on
 * a per mac end point basis, We have at most 1 such thread per mac end point
 * at a time.
 *
 * All other operations that are not serialized are essentially multi-threaded.
 * For example a control operation (get) like getting statistics which may not
 * care about reading values atomically or data threads sending or receiving
 * data. Mostly these type of operations don't modify the control state. Any
 * state these operations care about are protected using traditional locks.
 *
 * The perimeter only serializes serial operations. It does not imply there
 * aren't any other concurrent operations. However a serialized operation may
 * sometimes need to make sure it is the only thread. In this case it needs
 * to use reference counting mechanisms to cv_wait until any current data
 * threads are done.
 *
 * The mac layer itself does not hold any locks across a call to another layer.
 * The perimeter is however held across a down call to the driver to make the
 * whole control operation atomic with respect to other control operations.
 * Also the data path and get type control operations may proceed concurrently.
 * These operations synchronize with the single serial operation on a given mac
```



```
 * end point using regular locks. The perimeter ensures that conflicting
 * operations like say a mac_multicast_add and a mac_multicast_remove on the
 * same mac end point don't interfere with each other and also ensures that the
 * changes in the mac layer and the call to the underlying driver to say add a
 * multicast address are done atomically without interference from a thread
 * trying to delete the same address.
 *
 * For example, consider
 * mac_multicst_add()
 * {
 *      mac_perimeter_enter();  serialize all control operations
 *
 *      grab list lock          protect against access by data threads
 *      add to list
 *      drop list lock
 *
 *      call driver's mi_multicst
 *
 *      mac_perimeter_exit();
 * }
 *
 * To lessen the number of serialization locks and simplify the lock hierarchy,
 * we serialize all the control operations on a per mac end point by using a
 * single serialization lock called the perimeter. We allow recursive entry into
 * the perimeter to facilitate use of this mechanism by both the mac client and
 * the MAC layer itself.
 *
 * MAC client means an entity that does an operation on a mac handle
 * obtained from a mac_open/mac_client_open. Similarly MAC driver means
 * an entity that does an operation on a mac handle obtained from a
 * mac_register. An entity could be both client and driver but on different
 * handles eg. aggr. and should only make the corresponding mac interface calls
 * i.e. mac driver interface or mac client interface as appropriate for that
 * mac handle.
 *
 * General rules.
 * --------------
 *
 * R1. The lock order of upcall threads is natually opposite to downcall
 * threads. Hence upcalls must not hold any locks across layers for fear of
 * recursive lock enter and lock order violation. This applies to all layers.
 *
 * R2. The perimeter is just another lock. Since it is held in the down
 * direction, acquiring the perimeter in an upcall is prohibited as it would
 * cause a deadlock. This applies to all layers.
 *
 * Note that upcalls that need to grab the mac perimeter (for example
 * mac_notify upcalls) can still achieve that by posting the request to a
 * thread, which can then grab all the required perimeters and locks in the
 * right global order. Note that in the above example the mac layer iself
 * won't grab the mac perimeter in the mac_notify upcall, instead the upcall
 * to the client must do that. Please see the aggr code for an example.
 *
 * MAC client rules
 * ----------------
 *
 * R3. A MAC client may use the MAC provided perimeter facility to serialize
 * control operations on a per mac end point. It does this by by acquring
 * and holding the perimeter across a sequence of calls to the mac layer.
 * This ensures atomicity across the entire block of mac calls. In this
 * model the MAC client must not hold any client locks across the calls to
 * the mac layer. This model is the preferred solution.
 *
 * R4. However if a MAC client has a lot of global state across all mac end
 * points the per mac end point serialization may not be sufficient. In this
 * case the client may choose to use global locks or use its own serialization.
```

```
126  * To avoid deadlocks, these client layer locks held across the mac calls
127  * in the control path must never be acquired by the data path for the reason
128  * mentioned below.
129  *
130  * (Assume that a control operation that holds a client lock blocks in the
131  * mac layer waiting for upcall reference counts to drop to zero. If an upcall
132  * data thread that holds this reference count, tries to acquire the same
133  * client lock subsequently it will deadlock).
134  *
135  * A MAC client may follow either the R3 model or the R4 model, but can't
136  * mix both. In the former, the hierarchy is Perim -> client locks, but in
137  * the latter it is client locks -> Perim.
138  *
139  * R5. MAC clients must make MAC calls (excluding data calls) in a cv_wait'able
140  * context since they may block while trying to acquire the perimeter.
141  * In addition some calls may block waiting for upcall refcnts to come down to
142  * zero.
143  *
144  * R6. MAC clients must make sure that they are single threaded and all threads
145  * from the top (in particular data threads) have finished before calling
146  * mac_client_close. The MAC framework does not track the number of client
147  * threads using the mac client handle. Also mac clients must make sure
148  * they have undone all the control operations before calling mac_client_close.
149  * For example mac_unicast_remove/mac_multicast_remove to undo the corresponding
150  * mac_unicast_add/mac_multicast_add.
151  *
152  * MAC framework rules
153  * -------------------
154  *
155  * R7. The mac layer itself must not hold any mac layer locks (except the mac
156  * perimeter) across a call to any other layer from the mac layer. The call to
157  * any other layer could be via mi_* entry points, classifier entry points into
158  * the driver or via upcall pointers into layers above. The mac perimeter may
159  * be acquired or held only in the down direction, for e.g. when calling into
160  * a mi_* driver enty point to provide atomicity of the operation.
161  *
162  * R8. Since it is not guaranteed (see R14) that drivers won't hold locks across
163  * mac driver interfaces, the MAC layer must provide a cut out for control
164  * interfaces like upcall notifications and start them in a separate thread.
165  *
166  * R9. Note that locking order also implies a plumbing order. For example
167  * VNICs are allowed to be created over aggrs, but not vice-versa. An attempt
168  * to plumb in any other order must be failed at mac_open time, otherwise it
169  * could lead to deadlocks due to inverse locking order.
170  *
171  * R10. MAC driver interfaces must not block since the driver could call them
172  * in interrupt context.
173  *
174  * R11. Walkers must preferably not hold any locks while calling walker
175  * callbacks. Instead these can operate on reference counts. In simple
176  * callbacks it may be ok to hold a lock and call the callbacks, but this is
177  * harder to maintain in the general case of arbitrary callbacks.
178  *
179  * R12. The MAC layer must protect upcall notification callbacks using reference
180  * counts rather than holding locks across the callbacks.
181  *
182  * R13. Given the variety of drivers, it is preferable if the MAC layer can make
183  * sure that any pointers (such as mac ring pointers) it passes to the driver
184  * remain valid until mac unregister time. Currently the mac layer achieves
185  * this by using generation numbers for rings and freeing the mac rings only
186  * at unregister time.  The MAC layer must provide a layer of indirection and
187  * must not expose underlying driver rings or driver data structures/pointers
188  * directly to MAC clients.
189  *
190  * MAC driver rules
191  * ----------------
```



```
192  *
193  * R14. It would be preferable if MAC drivers don't hold any locks across any
194  * mac call. However at a minimum they must not hold any locks across data
195  * upcalls. They must also make sure that all references to mac data structures
196  * are cleaned up and that it is single threaded at mac_unregister time.
197  *
198  * R15. MAC driver interfaces don't block and so the action may be done
199  * asynchronously in a separate thread as for example handling notifications.
200  * The driver must not assume that the action is complete when the call
201  * returns.
202  *
203  * R16. Drivers must maintain a generation number per Rx ring, and pass it
204  * back to mac_rx_ring(); They are expected to increment the generation
205  * number whenever the ring's stop routine is invoked.
206  * See comments in mac_rx_ring();
207  *
208  * R17 Similarly mi_stop is another synchronization point and the driver must
209  * ensure that all upcalls are done and there won't be any future upcall
210  * before returning from mi_stop.
211  *
212  * R18. The driver may assume that all set/modify control operations via
213  * the mi_* entry points are single threaded on a per mac end point.
214  *
215  * Lock and Perimeter hierarchy scenarios
216  * ---------------------------------------
217  *
218  * i_mac_impl_lock -> mi_rw_lock -> srs_lock -> s_ring_lock[i_mac_tx_srs_notify]
219  *
220  * ft_lock -> fe_lock [mac_flow_lookup]
221  *
222  * mi_rw_lock -> fe_lock [mac_bcast_send]
223  *
224  * srs_lock -> mac_bw_lock [mac_rx_srs_drain_bw]
225  *
226  * cpu_lock -> mac_srs_g_lock -> srs_lock -> s_ring_lock [mac_walk_srs_and_bind]
227  *
228  * i_dls_devnet_lock -> mac layer locks [dls_devnet_rename]
229  *
230  * Perimeters are ordered P1 -> P2 -> P3 from top to bottom in order of mac
231  * client to driver. In the case of clients that explictly use the mac provided
232  * perimeter mechanism for its serialization, the hierarchy is
233  * Perimeter -> mac layer locks, since the client never holds any locks across
234  * the mac calls. In the case of clients that use its own locks the hierarchy
235  * is Client locks -> Mac Perim -> Mac layer locks. The client never explicitly
236  * calls mac_perim_enter/exit in this case.
237  *
238  * Subflow creation rules
239  * ----------------------------
240  * o In case of a user specified cpulist present on underlying link and flows,
241  * the flows cpulist must be a subset of the underlying link.
242  * o In case of a user specified fanout mode present on link and flow, the
243  * subflow fanout count has to be less than or equal to that of the
244  * underlying link. The cpu-bindings for the subflows will be a subset of
245  * the underlying link.
246  * o In case if no cpulist specified on both underlying link and flow, the
247  * underlying link relies on a  MAC tunable to provide out of box fanout.
248  * The subflow will have no cpulist (the subflow will be unbound)
249  * o In case if no cpulist is specified on the underlying link, a subflow can
250  * carry  either a user-specified cpulist or fanout count. The cpu-bindings
251  * for the subflow will not adhere to restriction that they need to be subset
252  * of the underlying link.
253  * o In case where the underlying link is carrying either a user specified
254  * cpulist or fanout mode and for a unspecified subflow, the subflow will be
255  * created unbound.
256  * o While creating unbound subflows, bandwidth mode changes attempt to
257  * figure a right fanout count. In such cases the fanout count will override
```

```
 258  * the unbound cpu-binding behavior.
 259  * o In addition to this, while cycling between flow and link properties, we
 260  * impose a restriction that if a link property has a subflow with
 261  * user-specified attributes, we will not allow changing the link property.
 262  * The administrator needs to reset all the user specified properties for the
 263  * subflows before attempting a link property change.
 264  * Some of the above rules can be overridden by specifying additional command
 265  * line options while creating or modifying link or subflow properties.
 266  */

 268 #include <sys/types.h>
 269 #include <sys/conf.h>
 270 #include <sys/id_space.h>
 271 #include <sys/esunddi.h>
 272 #include <sys/stat.h>
 273 #include <sys/mkdev.h>
 274 #include <sys/stream.h>
 275 #include <sys/strsun.h>
 276 #include <sys/strsubr.h>
 277 #include <sys/dlpi.h>
 278 #include <sys/list.h>
 279 #include <sys/modhash.h>
 280 #include <sys/mac_provider.h>
 281 #include <sys/mac_client_impl.h>
 282 #include <sys/mac_soft_ring.h>
 283 #include <sys/mac_stat.h>
 284 #include <sys/mac_impl.h>
 285 #include <sys/mac.h>
 286 #include <sys/dls.h>
 287 #include <sys/dld.h>
 288 #include <sys/modctl.h>
 289 #include <sys/fs/dv_node.h>
 290 #include <sys/thread.h>
 291 #include <sys/proc.h>
 292 #include <sys/callb.h>
 293 #include <sys/cpuvar.h>
 294 #include <sys/atomic.h>
 295 #include <sys/bitmap.h>
 296 #include <sys/sdt.h>
 297 #include <sys/mac_flow.h>
 298 #include <sys/ddi_intr_impl.h>
 299 #include <sys/disp.h>
 300 #include <sys/sdt.h>
 301 #include <sys/vnic.h>
 302 #include <sys/vnic_impl.h>
 303 #include <sys/vlan.h>
 304 #include <inet/ip.h>
 305 #include <inet/ip6.h>
 306 #include <sys/exacct.h>
 307 #include <sys/exacct_impl.h>
 308 #include <inet/nd.h>
 309 #include <sys/ethernet.h>
 310 #include <sys/pool.h>
 311 #include <sys/pool_pset.h>
 312 #include <sys/cpupart.h>
 313 #include <inet/wifi_ioctl.h>
 314 #include <net/wpa.h>

 315 #define IMPL_HASHSZ     67      /* prime */

 317 kmem_cache_t            *i_mac_impl_cachep;
 318 mod_hash_t              *i_mac_impl_hash;
 319 krwlock_t               i_mac_impl_lock;
 320 uint_t                  i_mac_impl_count;
 321 static kmem_cache_t     *mac_ring_cache;
 322 static id_space_t       *minor_ids;
```



```
 323 static uint32_t         minor_count;
 324 static pool_event_cb_t  mac_pool_event_reg;

 326 /*
 327  * Logging stuff. Perhaps mac_logging_interval could be broken into
 328  * mac_flow_log_interval and mac_link_log_interval if we want to be
 329  * able to schedule them differently.
 330  */
 331 uint_t                  mac_logging_interval;
 332 boolean_t               mac_flow_log_enable;
 333 boolean_t               mac_link_log_enable;
 334 timeout_id_t            mac_logging_timer;

 336 /* for debugging, see MAC_DBG_PRT() in mac_impl.h */
 337 int mac_dbg = 0;

 339 #define MACTYPE_KMODDIR "mac"
 340 #define MACTYPE_HASHSZ  67
 341 static mod_hash_t       *i_mactype_hash;
 342 /*
 343  * i_mactype_lock synchronizes threads that obtain references to mactype_t
 344  * structures through i_mactype_getplugin().
 345  */
 346 static kmutex_t         i_mactype_lock;

 348 /*
 349  * mac_tx_percpu_cnt
 350  *
 351  * Number of per cpu locks per mac_client_impl_t. Used by the transmit side
 352  * in mac_tx to reduce lock contention. This is sized at boot time in mac_init.
 353  * mac_tx_percpu_cnt_max is settable in /etc/system and must be a power of 2.
 354  * Per cpu locks may be disabled by setting mac_tx_percpu_cnt_max to 1.
 355  */
 356 int mac_tx_percpu_cnt;
 357 int mac_tx_percpu_cnt_max = 128;

 359 /*
 360  * Call back functions for the bridge module.  These are guaranteed to be valid
 361  * when holding a reference on a link or when holding mip->mi_bridge_lock and
 362  * mi_bridge_link is non-NULL.
 363  */
 364 mac_bridge_tx_t mac_bridge_tx_cb;
 365 mac_bridge_rx_t mac_bridge_rx_cb;
 366 mac_bridge_ref_t mac_bridge_ref_cb;
 367 mac_bridge_ls_t mac_bridge_ls_cb;

 369 static int i_mac_constructor(void *, void *, int);
 370 static void i_mac_destructor(void *, void *);
 371 static int i_mac_ring_ctor(void *, void *, int);
 372 static void i_mac_ring_dtor(void *, void *);
 373 static mblk_t *mac_rx_classify(mac_impl_t *, mac_resource_handle_t, mblk_t *);
 374 void mac_tx_client_flush(mac_client_impl_t *);
 375 void mac_tx_client_block(mac_client_impl_t *);
 376 static void mac_rx_ring_quiesce(mac_ring_t *, uint_t);
 377 static int mac_start_group_and_rings(mac_group_t *);
 378 static void mac_stop_group_and_rings(mac_group_t *);
 379 static void mac_pool_event_cb(pool_event_t, int, void *);

 381 typedef struct netinfo_s {
 382         list_node_t     ni_link;
 383         void            *ni_record;
 384         int             ni_size;
 385         int             ni_type;
 386 } netinfo_t;
```

*unchanged_portion_omitted_*

```
2845 /*
2846  * Checks the size of the value size specified for a property as
2847  * part of a property operation. Returns B_TRUE if the size is
2848  * correct, B_FALSE otherwise.
2849  */
2850 boolean_t
2851 mac_prop_check_size(mac_prop_id_t id, uint_t valsize, boolean_t is_range)
2852 {
2853         uint_t minsize = 0;

2855         if (is_range)
2856                 return (valsize >= sizeof (mac_propval_range_t));

2858         switch (id) {
2859         case MAC_PROP_ZONE:
2860                 minsize = sizeof (dld_ioc_zid_t);
2861                 break;
2862         case MAC_PROP_AUTOPUSH:
2863                 if (valsize != 0)
2864                         minsize = sizeof (struct dlautopush);
2865                 break;
2866         case MAC_PROP_TAGMODE:
2867                 minsize = sizeof (link_tagmode_t);
2868                 break;
2869         case MAC_PROP_RESOURCE:
2870         case MAC_PROP_RESOURCE_EFF:
2871                 minsize = sizeof (mac_resource_props_t);
2872                 break;
2873         case MAC_PROP_DUPLEX:
2874                 minsize = sizeof (link_duplex_t);
2875                 break;
2876         case MAC_PROP_SPEED:
2877                 minsize = sizeof (uint64_t);
2878                 break;
2879         case MAC_PROP_STATUS:
2880                 minsize = sizeof (link_state_t);
2881                 break;
2882         case MAC_PROP_AUTONEG:
2883         case MAC_PROP_EN_AUTONEG:
2884                 minsize = sizeof (uint8_t);
2885                 break;
2886         case MAC_PROP_MTU:
2887         case MAC_PROP_LLIMIT:
2888         case MAC_PROP_LDECAY:
2889                 minsize = sizeof (uint32_t);
2890                 break;
2891         case MAC_PROP_FLOWCTRL:
2892                 minsize = sizeof (link_flowctrl_t);
2893                 break;
2894         case MAC_PROP_ADV_10GFDX_CAP:
2895         case MAC_PROP_EN_10GFDX_CAP:
2896         case MAC_PROP_ADV_1000HDX_CAP:
2897         case MAC_PROP_EN_1000HDX_CAP:
2898         case MAC_PROP_ADV_100FDX_CAP:
2899         case MAC_PROP_EN_100FDX_CAP:
2900         case MAC_PROP_ADV_100HDX_CAP:
2901         case MAC_PROP_EN_100HDX_CAP:
2902         case MAC_PROP_ADV_10FDX_CAP:
2903         case MAC_PROP_EN_10FDX_CAP:
2904         case MAC_PROP_ADV_10HDX_CAP:
2905         case MAC_PROP_EN_10HDX_CAP:
2906         case MAC_PROP_ADV_100T4_CAP:
2907         case MAC_PROP_EN_100T4_CAP:
2908                 minsize = sizeof (uint8_t);
2909                 break;
2910         case MAC_PROP_PVID:
```



```
2911                 minsize = sizeof (uint16_t);
2912                 break;
2913         case MAC_PROP_IPTUN_HOPLIMIT:
2914                 minsize = sizeof (uint32_t);
2915                 break;
2916         case MAC_PROP_IPTUN_ENCAPLIMIT:
2917                 minsize = sizeof (uint32_t);
2918                 break;
2919         case MAC_PROP_MAX_TX_RINGS_AVAIL:
2920         case MAC_PROP_MAX_RX_RINGS_AVAIL:
2921         case MAC_PROP_MAX_RXHWCLNT_AVAIL:
2922         case MAC_PROP_MAX_TXHWCLNT_AVAIL:
2923                 minsize = sizeof (uint_t);
2924                 break;
2925         case MAC_PROP_WL_ESSID:
2926                 minsize = sizeof (wl_linkstatus_t);
2927                 break;
2928         case MAC_PROP_WL_BSSID:
2929                 minsize = sizeof (wl_bssid_t);
2930                 break;
2931         case MAC_PROP_WL_BSSTYPE:
2932                 minsize = sizeof (wl_bss_type_t);
2933                 break;
2934         case MAC_PROP_WL_LINKSTATUS:
2935                 minsize = sizeof (wl_linkstatus_t);
2936                 break;
2937         case MAC_PROP_WL_DESIRED_RATES:
2938                 minsize = sizeof (wl_rates_t);
2939                 break;
2940         case MAC_PROP_WL_SUPPORTED_RATES:
2941                 minsize = sizeof (wl_rates_t);
2942                 break;
2943         case MAC_PROP_WL_AUTH_MODE:
2944                 minsize = sizeof (wl_authmode_t);
2945                 break;
2946         case MAC_PROP_WL_ENCRYPTION:
2947                 minsize = sizeof (wl_encryption_t);
2948                 break;
2949         case MAC_PROP_WL_RSSI:
2950                 minsize = sizeof (wl_rssi_t);
2951                 break;
2952         case MAC_PROP_WL_PHY_CONFIG:
2953                 minsize = sizeof (wl_phy_conf_t);
2954                 break;
2955         case MAC_PROP_WL_CAPABILITY:
2956                 minsize = sizeof (wl_capability_t);
2957                 break;
2958         case MAC_PROP_WL_WPA:
2959                 minsize = sizeof (wl_wpa_t);
2960                 break;
2962         case MAC_PROP_WL_SCANRESULTS:
2963                 minsize = sizeof (wl_wpa_ess_t);
2964                 break;
2961         case MAC_PROP_WL_POWER_MODE:
2962                 minsize = sizeof (wl_ps_mode_t);
2963                 break;
2964         case MAC_PROP_WL_RADIO:
2965                 minsize = sizeof (wl_radio_t);
2966                 break;
2967         case MAC_PROP_WL_ESS_LIST:
2968                 minsize = sizeof (wl_ess_list_t);
2969                 break;
2970         case MAC_PROP_WL_KEY_TAB:
2971                 minsize = sizeof (wl_wep_key_tab_t);
2972                 break;
2973         case MAC_PROP_WL_CREATE_IBSS:
```

```
2974                 minsize = sizeof (wl_create_ibss_t);
2975                 break;
2976         case MAC_PROP_WL_SETOPTIE:
2977                 minsize = sizeof (wl_wpa_ie_t);
2978                 break;
2979         case MAC_PROP_WL_DELKEY:
2980                 minsize = sizeof (wl_del_key_t);
2981                 break;
2982         case MAC_PROP_WL_KEY:
2983                 minsize = sizeof (wl_key_t);
2984                 break;
2985         case MAC_PROP_WL_MLME:
2986                 minsize = sizeof (wl_mlme_t);
2987                 break;
2988         }

2990         return (valsize >= minsize);
2991 }
```

*unchanged_portion_omitted_*

****
23381 Tue Jun 12 19:55:23 2012
new/usr/src/uts/common/io/net80211/net80211.c
updated libdladm public interface
prevented usr.lib/wpa_supplicant from being checked with lint
****
_____***unchanged_portion_omitted_***

```
 188 /*
 189  * Notify state transition event message to WPA daemon
 190  */
 191 void
 192 ieee80211_notify(ieee80211com_t *ic, wpa_event_type event)
 193 {
 194         if ((ic->ic_flags & IEEE80211_F_WPA) == 0)
 195                 return;         /* Not running on WPA mode */

 194         ic->ic_eventq[ic->ic_evq_tail] = event;
 195         ic->ic_evq_tail ++;
 196         if (ic->ic_evq_tail >= MAX_EVENT) ic->ic_evq_tail = 0;

 198         /* async */
 199         (void) timeout(ieee80211_event_thread, (void *)ic, 0);
 200 }
```

_____***unchanged_portion_omitted_***

```
**********************************************************
   57762 Tue Jun 12 19:55:24 2012
new/usr/src/uts/common/io/net80211/net80211_ioctl.c
ess_list ioctl now provides all scan results properties for wpa/libdlwlan
first integration of wpa_s control interface client code
first integration of wpa_s wpa_ie parsing code
**********************************************************
```

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

/*
 * Redistribution and use in source and binary forms, with or without
 * modification, are permitted provided that the following conditions
 * are met:
 * 1. Redistributions of source code must retain the above copyright
 *    notice, this list of conditions and the following disclaimer.
 * 2. 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.
 * 3. The name of the author may not be used to endorse or promote products
 *    derived from this software without specific prior written permission.
 *
 * Alternatively, this software may be distributed under the terms of the
 * GNU General Public License ("GPL") version 2 as published by the Free
 * Software Foundation.
 *
 * THIS SOFTWARE IS PROVIDED BY THE AUTHOR ‘‘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 AUTHOR 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.
 */


#include <sys/param.h>
#include <sys/types.h>
#include <sys/errno.h>
#include <sys/strsun.h>
#include <sys/policy.h>
#include <inet/common.h>
#include <inet/nd.h>
#include <inet/mi.h>
#include <sys/note.h>
#include <sys/mac_provider.h>
#include <inet/wifi_ioctl.h>
#include "net80211_impl.h"

static int wl_set_essid(struct ieee80211com *, const void *);
static void wl_get_essid(struct ieee80211com *, void *);
static int wl_set_bssid(struct ieee80211com *, const void *);
static void wl_get_bssid(struct ieee80211com *, void *);
static int wl_set_bsstype(struct ieee80211com *, const void *);
static void wl_get_bsstype(struct ieee80211com *, void *);
static void wl_get_linkstatus(struct ieee80211com *, void *);
static int wl_set_desrates(struct ieee80211com *, const void *);
static void wl_get_desrates(struct ieee80211com *, void *);
static int wl_set_authmode(struct ieee80211com *, const void *);
static void wl_get_authmode(struct ieee80211com *, void *);
static int wl_set_encrypt(struct ieee80211com *, const void *);
```



```
static void wl_get_encrypt(struct ieee80211com *, void *);
static void wl_get_rssi(struct ieee80211com *, void *);
static int wl_set_phy(struct ieee80211com *, const void *);
static int wl_get_phy(struct ieee80211com *, void *);
static void wl_get_capability(struct ieee80211com *, void *);
static int wl_set_wpa(struct ieee80211com *, const void *);
static void wl_get_wpa(struct ieee80211com *, void *);
static void wl_get_scanresults(struct ieee80211com *, void *);
static void wl_get_esslist(struct ieee80211com *, void *);
static int wl_set_wepkey(struct ieee80211com *, const void *);
static int wl_set_optie(struct ieee80211com *, const void *);
static int wl_set_delkey(struct ieee80211com *, const void *);
static int wl_set_mlme(struct ieee80211com *, const void *);
static int wl_set_wpakey(struct ieee80211com *, const void *);
static void wl_get_suprates(struct ieee80211com *, void *);
static int wl_set_createibss(struct ieee80211com *, const void *);
static void wl_get_createibss(struct ieee80211com *, void *);

static size_t
wifi_strnlen(const char *s, size_t n)
{
        size_t i;

        for (i = 0; i < n && s[i] != '\0'; i++)
                /* noop */;
        return (i);
}
```

*unchanged_portion_omitted_*

```
#define WIFI_HAVE_CAP(in, flag) (((in)->in_capinfo & (flag)) ? 1 : 0)
#define WIFI_HAVE_HTCAP(in)     (((in)->in_htcap != 0) ? 1 : 0)

/*
 * Callback function used by ieee80211_iterate_nodes() in
 * wifi_cfg_esslist() to get info of each node in a node table
 *    arg  output buffer, pointer to wl_ess_list_t
 *    in   each node in the node table
 */
static void
wifi_read_ap(void *arg, struct ieee80211_node *in)
{
        wl_ess_list_t *aps = arg;
        ieee80211com_t *ic = in->in_ic;
        struct ieee80211_channel *chan = in->in_chan;
        struct ieee80211_rateset *rates = &(in->in_rates);
        wl_ess_conf_t *conf;
        uint8_t *end;
        uint_t i, nrates;

        end = (uint8_t *)aps - WIFI_BUF_OFFSET + MAX_BUF_LEN -
            sizeof (wl_ess_list_t);
        conf = &aps->wl_ess_list_ess[aps->wl_ess_list_num];
        if ((uint8_t *)conf > end)
                return;

        conf->wl_ess_conf_length = sizeof (struct wl_ess_conf);

        /* skip newly allocated NULL bss node */
        if (IEEE80211_ADDR_EQ(in->in_macaddr, ic->ic_macaddr))
                return;

        conf->wl_ess_conf_essid.wl_essid_length = in->in_esslen;
        bcopy(in->in_essid, conf->wl_ess_conf_essid.wl_essid_essid,
            in->in_esslen);
        bcopy(in->in_bssid, conf->wl_ess_conf_bssid, IEEE80211_ADDR_LEN);
```

```
735         conf->wl_ess_conf_beacon_period = in->in_intval;
736         conf->wl_ess_conf_beacon_tsf = in->in_tstamp.tsf;
737         conf->wl_ess_conf_beacon_age = in->in_rstamp;

735         conf->wl_ess_conf_wepenabled =
736             (in->in_capinfo & IEEE80211_CAPINFO_PRIVACY ?
737             WL_ENC_WEP : WL_NOENCRYPTION);
738         conf->wl_ess_conf_bsstype =
739             (in->in_capinfo & IEEE80211_CAPINFO_ESS ?
740             WL_BSS_BSS : WL_BSS_IBSS);
739         conf->wl_ess_conf_sl = wifi_getrssi(in);

741         conf->wl_ess_conf_caps = in->in_capinfo;
742         /* authmode is set to IEEE80211_AUTH_OPEN = 1 by default */

744         if (in->in_wpa_ie == NULL) {
745                 conf->wl_ess_conf_wpa_ie_len = 0;
746         } else {
747                 conf->wl_ess_conf_wpa_ie_len = in->in_wpa_ie[1] + 2;
748                 bcopy(in->in_wpa_ie, conf->wl_ess_conf_wpa_ie,
749                     conf->wl_ess_conf_wpa_ie_len);
750         }
742         conf->wl_ess_conf_reserved[0] = (in->in_wpa_ie == NULL? 0 : 1);

752         /* physical (FH, DS, ERP) parameters */
753         if (IEEE80211_IS_CHAN_A(chan) || IEEE80211_IS_CHAN_T(chan)) {
754                 wl_ofdm_t *ofdm =
755                     (wl_ofdm_t *)&((conf->wl_phy_conf).wl_phy_ofdm_conf);
756                 ofdm->wl_ofdm_subtype = WL_OFDM;
757                 ofdm->wl_ofdm_frequency = chan->ich_freq;
758                 ofdm->wl_ofdm_ht_enabled = WIFI_HAVE_HTCAP(in);
759         } else {
760                 switch (in->in_phytype) {
761                 case IEEE80211_T_FH: {
762                         wl_fhss_t *fhss = (wl_fhss_t *)
763                             &((conf->wl_phy_conf).wl_phy_fhss_conf);

765                         fhss->wl_fhss_subtype = WL_FHSS;
766                         fhss->wl_fhss_channel = ieee80211_chan2ieee(ic, chan);
767                         fhss->wl_fhss_dwelltime = in->in_fhdwell;
768                         break;
769                 }
770                 case IEEE80211_T_DS: {
771                         wl_dsss_t *dsss = (wl_dsss_t *)
772                             &((conf->wl_phy_conf).wl_phy_dsss_conf);

774                         dsss->wl_dsss_subtype = WL_DSSS;
775                         dsss->wl_dsss_channel = ieee80211_chan2ieee(ic, chan);
776                         dsss->wl_dsss_have_short_preamble = WIFI_HAVE_CAP(in,
777                             IEEE80211_CAPINFO_SHORT_PREAMBLE);
778                         dsss->wl_dsss_agility_enabled = WIFI_HAVE_CAP(in,
779                             IEEE80211_CAPINFO_CHNL_AGILITY);
780                         dsss->wl_dsss_have_pbcc = dsss->wl_dsss_pbcc_enable =
781                             WIFI_HAVE_CAP(in, IEEE80211_CAPINFO_PBCC);
782                         break;
783                 }
784                 case IEEE80211_T_OFDM: {
785                         wl_erp_t *erp = (wl_erp_t *)
786                             &((conf->wl_phy_conf).wl_phy_erp_conf);

788                         erp->wl_erp_subtype = WL_ERP;
789                         erp->wl_erp_channel = ieee80211_chan2ieee(ic, chan);
790                         erp->wl_erp_have_short_preamble = WIFI_HAVE_CAP(in,
791                             IEEE80211_CAPINFO_SHORT_PREAMBLE);
792                         erp->wl_erp_have_agility = erp->wl_erp_agility_enabled =
793                             WIFI_HAVE_CAP(in, IEEE80211_CAPINFO_CHNL_AGILITY);
```



```
794                         erp->wl_erp_have_pbcc = erp->wl_erp_pbcc_enabled =
795                             WIFI_HAVE_CAP(in, IEEE80211_CAPINFO_PBCC);
796                         erp->wl_erp_dsss_ofdm_enabled =
797                             WIFI_HAVE_CAP(in, IEEE80211_CAPINFO_DSSSOFDM);
798                         erp->wl_erp_sst_enabled = WIFI_HAVE_CAP(in,
799                             IEEE80211_CAPINFO_SHORT_SLOTTIME);
800                         erp->wl_erp_ht_enabled = WIFI_HAVE_HTCAP(in);
801                         break;
802                 } /* case IEEE80211_T_OFDM */
803                 } /* switch in->in_phytype */
804         }

806         /* supported rates */
807         nrates = MIN(rates->ir_nrates, MAX_SCAN_SUPPORT_RATES);
808         /*
809          * The number of supported rates might exceed
810          * MAX_SCAN_SUPPORT_RATES. Fill in highest rates
811          * first so userland command could properly show
812          * maximum speed of AP
813          */
814         for (i = 0; i < nrates; i++) {
815                 conf->wl_supported_rates[i] =
816                     rates->ir_rates[rates->ir_nrates - i - 1];
817         }

819         aps->wl_ess_list_num++;
820 }
```

_____***unchanged_portion_omitted_***

```
1139 /*
1132  * To be compatible with drivers/tools of OpenSolaris.org,
1133  * we use a different ID to filter out those APs of WPA mode.
1134  */
1135 static int
1136 wifi_cfg_scanresults(struct ieee80211com *ic, uint32_t cmd, mblk_t **mp)
1137 {
1138         mblk_t *omp;
1139         wldp_t *outp;
1140         wl_wpa_ess_t *sr;
1141         ieee80211_node_t *in;
1142         ieee80211_node_table_t *nt;
1143         int len, ap_num = 0;
1144         int err = 0;

1146         if ((omp = wifi_getoutmsg(*mp, cmd, MAX_BUF_LEN - WIFI_BUF_OFFSET)) ==
1147             NULL) {
1148                 return (ENOMEM);
1149         }
1150         outp = (wldp_t *)omp->b_rptr;
1151         sr = (wl_wpa_ess_t *)outp->wldp_buf;
1152         sr->count = 0;

1154         switch (cmd) {
1155         case WLAN_GET_PARAM:
1156                 ieee80211_dbg(IEEE80211_MSG_WPA, "wifi_cfg_scanresults\n");
1157                 nt = &ic->ic_scan;
1158                 IEEE80211_NODE_LOCK(nt);
1159                 in = list_head(&nt->nt_node);
1160                 while (in != NULL) {
1161                         /* filter out non-WPA APs */
1162                         if (in->in_wpa_ie == NULL) {
1163                                 in = list_next(&nt->nt_node, in);
1164                                 continue;
1165                         }
1166                         bcopy(in->in_bssid, sr->ess[ap_num].bssid,
1167                             IEEE80211_ADDR_LEN);
```

```
1168                        sr->ess[ap_num].ssid_len = in->in_esslen;
1169                        bcopy(in->in_essid, sr->ess[ap_num].ssid,
1170                            in->in_esslen);
1171                        sr->ess[ap_num].freq = in->in_chan->ich_freq;

1173                        len = in->in_wpa_ie[1] + 2;
1174                        bcopy(in->in_wpa_ie, sr->ess[ap_num].wpa_ie, len);
1175                        sr->ess[ap_num].wpa_ie_len = len;

1177                        ap_num ++;
1178                        in = list_next(&nt->nt_node, in);
1179                }
1180                IEEE80211_NODE_UNLOCK(nt);
1181                sr->count = ap_num;
1182                outp->wldp_length = WIFI_BUF_OFFSET +
1183                    offsetof(wl_wpa_ess_t, ess) +
1184                    sr->count * sizeof (struct wpa_ess);
1185                omp->b_wptr = omp->b_rptr + outp->wldp_length;
1186                break;
1187        case WLAN_SET_PARAM:
1188                outp->wldp_result = WL_READONLY;
1189                err = EINVAL;
1190                break;
1191        default:
1192                ieee80211_err("wifi_cfg_scanresults: unknown cmmand %x\n", cmd);
1193                outp->wldp_result = WL_NOTSUPPORTED;
1194                err = EINVAL;
1195                break;
1196        }

1198        freemsg(*mp);
1199        *mp = omp;
1200        return (err);
1201 }

1203 /*
1140  * Manually control the state of AUTH | DEAUTH | DEASSOC | ASSOC
1141  */
1142 static int
1143 wifi_cfg_setmlme(struct ieee80211com *ic, uint32_t cmd, mblk_t **mp)
1144 {
1145         mblk_t *omp;
1146         wldp_t *outp;
1147         wldp_t *inp = (wldp_t *)(*mp)->b_rptr;
1148         wl_mlme_t *mlme = (wl_mlme_t *)inp->wldp_buf;
1149         int err = 0;

1151         if ((omp = wifi_getoutmsg(*mp, cmd, 0)) == NULL)
1152                 return (ENOMEM);
1153         outp = (wldp_t *)omp->b_rptr;

1155         switch (cmd) {
1156         case WLAN_GET_PARAM:
1157                 outp->wldp_result = WL_WRITEONLY;
1158                 err = EINVAL;
1159                 break;
1160         case WLAN_SET_PARAM:
1161                 err = wl_set_mlme(ic, mlme);
1162                 break;
1163         default:
1164                 ieee80211_err("wifi_cfg_delkey: unknown command %x\n", cmd);
1165                 outp->wldp_result = WL_NOTSUPPORTED;
1166                 err = EINVAL;
1167                 break;
1168         }
```



```
1170         freemsg(*mp);
1171         *mp = omp;
1172         return (err);
1173 }

1175 static int
1176 wifi_cfg_getset(struct ieee80211com *ic, mblk_t **mp, uint32_t cmd)
1177 {
1178         mblk_t *mp1 = *mp;
1179         wldp_t *wp = (wldp_t *)mp1->b_rptr;
1180         int err = 0;

1182         ASSERT(ic != NULL && mp1 != NULL);
1183         IEEE80211_LOCK_ASSERT(ic);
1184         if (MBLKL(mp1) < WIFI_BUF_OFFSET) {
1185                 ieee80211_err("wifi_cfg_getset: "
1186                     "invalid input buffer, size=%d\n", MBLKL(mp1));
1187                 return (EINVAL);
1188         }

1190         switch (wp->wldp_id) {
1191         /* Commands */
1192         case WL_SCAN:
1193                 err = wifi_cmd_scan(ic, mp1);
1194                 break;
1195         case WL_LOAD_DEFAULTS:
1196                 err = wifi_cmd_loaddefaults(ic, mp1);
1197                 break;
1198         case WL_DISASSOCIATE:
1199                 err = wifi_cmd_disassoc(ic, mp1);
1200                 break;
1201         /* Parameters */
1202         case WL_ESSID:
1203                 err = wifi_cfg_essid(ic, cmd, mp);
1204                 break;
1205         case WL_BSSID:
1206                 err = wifi_cfg_bssid(ic, cmd, mp);
1207                 break;
1208         case WL_NODE_NAME:
1209                 err = wifi_cfg_nodename(ic, cmd, mp);
1210                 break;
1211         case WL_PHY_CONFIG:
1212                 err = wifi_cfg_phy(ic, cmd, mp);
1213                 break;
1214         case WL_WEP_KEY_TAB:
1215                 err = wifi_cfg_wepkey(ic, cmd, mp);
1216                 break;
1217         case WL_WEP_KEY_ID:
1218                 err = wifi_cfg_keyid(ic, cmd, mp);
1219                 break;
1220         case WL_AUTH_MODE:
1221                 err = wifi_cfg_authmode(ic, cmd, mp);
1222                 break;
1223         case WL_ENCRYPTION:
1224                 err = wifi_cfg_encrypt(ic, cmd, mp);
1225                 break;
1226         case WL_BSS_TYPE:
1227                 err = wifi_cfg_bsstype(ic, cmd, mp);
1228                 break;
1229         case WL_CREATE_IBSS:
1230                 err = wifi_cfg_createibss(ic, cmd, mp);
1231                 break;
1232         case WL_DESIRED_RATES:
1233                 err = wifi_cfg_desrates(ic, cmd, mp);
1234                 break;
1235         case WL_LINKSTATUS:
```

```
1236                 err = wifi_cfg_linkstatus(ic, cmd, mp);
1237                 break;
1238         case WL_ESS_LIST:
1239                 err = wifi_cfg_esslist(ic, cmd, mp);
1240                 break;
1241         case WL_SUPPORTED_RATES:
1242                 err = wifi_cfg_suprates(ic, cmd, mp);
1243                 break;
1244         case WL_RSSI:
1245                 err = wifi_cfg_rssi(ic, cmd, mp);
1246                 break;
1311         /*
1312          * WPA IOCTLs
1313          */
1314         case WL_CAPABILITY:
1315                 err = wifi_cfg_caps(ic, cmd, mp);
1316                 break;
1317         case WL_WPA:
1318                 err = wifi_cfg_wpa(ic, cmd, mp);
1319                 break;
1320         case WL_KEY:
1321                 err = wifi_cfg_wpakey(ic, cmd, mp);
1322                 break;
1323         case WL_DELKEY:
1324                 err = wifi_cfg_delkey(ic, cmd, mp);
1325                 break;
1326         case WL_SETOPTIE:
1327                 err = wifi_cfg_setoptie(ic, cmd, mp);
1328                 break;
1329         case WL_SCANRESULTS:
1330                 err = wifi_cfg_scanresults(ic, cmd, mp);
1331                 break;
1332         case WL_MLME:
1333                 err = wifi_cfg_setmlme(ic, cmd, mp);
1334                 break;
1247         default:
1248                 wifi_setupoutmsg(mp1, 0);
1249                 wp->wldp_result = WL_LACK_FEATURE;
1250                 err = ENOTSUP;
1251                 break;
1252         }

1254         return (err);
1255 }
```

_____unchanged_portion_omitted_

```
1384 static void
1385 wl_get_essid(struct ieee80211com *ic, void *wldp_buf)
1386 {
1387         char *essid;
1388         wl_essid_t ow_essid;

1390         essid = (char *)ic->ic_des_essid;
1391         if (essid[0] == '\0')
1392                 essid = (char *)ic->ic_bss->in_essid;

1394         bzero(&ow_essid, sizeof (wl_essid_t));
1395         ow_essid.wl_essid_length = wifi_strnlen((const char *)essid,
1396             IEEE80211_NWID_LEN);
1397         bcopy(essid, ow_essid.wl_essid_essid,
1398             ow_essid.wl_essid_length);
1399         bcopy(&ow_essid, wldp_buf, sizeof (wl_essid_t));

1400 }
```

_____unchanged_portion_omitted_


```
1468 static void
1469 wl_get_bsstype(struct ieee80211com *ic, void *wldp_buf)
1470 {
1471         wl_bss_type_t ow_opmode;

1473         switch (ic->ic_opmode) {
1474         case IEEE80211_M_STA:
1475                 ow_opmode = WL_BSS_BSS;
1476                 break;
1477         case IEEE80211_M_IBSS:
1478                 ow_opmode = WL_BSS_IBSS;
1479                 break;
1480         default:
1481                 ow_opmode = WL_BSS_BSS;
1570                 ow_opmode = WL_BSS_ANY;
1482                 break;
1483         }

1485         bcopy(&ow_opmode, wldp_buf, sizeof (wl_bss_type_t));
1486 }
```

_____unchanged_portion_omitted_

```
1862 /*
1952  * MAC_PROP_WL_SCANRESULTS
1953  */

1955 static void
1956 wl_get_scanresults(struct ieee80211com *ic, void *wldp_buf)
1957 {
1958         wl_wpa_ess_t *sr;
1959         ieee80211_node_t *in;
1960         ieee80211_node_table_t *nt;
1961         int ap_num;
1962         int len;

1964         sr = (wl_wpa_ess_t *)wldp_buf;
1965         sr->count = 0;
1966         ap_num = 0;

1968         ieee80211_dbg(IEEE80211_MSG_WPA, "wl_get_scanrelults\n");

1970         nt = &ic->ic_scan;
1971         IEEE80211_NODE_LOCK(nt);
1972         in = list_head(&nt->nt_node);

1974         while (in != NULL) {
1975                 /* filter out non-wpa APs */
1976                 if (in->in_wpa_ie == NULL) {
1977                         in = list_next(&nt->nt_node, in);
1978                         continue;
1979                 }
1980                 bcopy(in->in_bssid, sr->ess[ap_num].bssid,
1981                     IEEE80211_ADDR_LEN);
1982                 sr->ess[ap_num].ssid_len = in->in_esslen;
1983                 bcopy(in->in_essid, sr->ess[ap_num].ssid,
1984                     in->in_esslen);
1985                 sr->ess[ap_num].freq = in->in_chan->ich_freq;

1987                 len = in->in_wpa_ie[1] + 2;
1988                 bcopy(in->in_wpa_ie, sr->ess[ap_num].wpa_ie, len);
1989                 sr->ess[ap_num].wpa_ie_len = len;

1991                 ap_num++;
1992                 in = list_next(&nt->nt_node, in);
1993         }
1994         IEEE80211_NODE_UNLOCK(nt);
```

```
1995         sr->count = ap_num;

1997 }

1999 /*
1863  * MAC_PROP_WL_ESS_LIST
1864  */
1865 static void
1866 wl_get_esslist(struct ieee80211com *ic, void *wldp_buf)
1867 {
1868         wl_ess_list_t *ess_list;

1870         ess_list = (wl_ess_list_t *)wldp_buf;

1872         ess_list->wl_ess_list_num = 0;
1873         ieee80211_iterate_nodes(&ic->ic_scan, wifi_read_ap, ess_list);

1875 }
______unchanged_portion_omitted_

1935 /*
1936  * MAC_PROP_WL_SETOPTIE
1937  */
1938 static int
1939 wl_set_optie(struct ieee80211com *ic, const void *wldp_buf)
1940 {
1941         int err = 0;
1942         char *ie;
1943         wl_wpa_ie_t *ie_in = (wl_wpa_ie_t *)wldp_buf;

1945         if (ic->ic_opmode != IEEE80211_M_STA) {
1946                 ieee80211_err("wl_set_optie: opmode err\n");
1947                 err = EINVAL;
1948                 return (err);
1949         }
1950         if (ie_in->wpa_ie_len > IEEE80211_MAX_OPT_IE) {

1952                 ieee80211_err("wl_set_optie: optie is too long\n");

1954                 err = EINVAL;
1955                 return (err);
1956         }

1958         ie = ieee80211_malloc(ie_in->wpa_ie_len);
1959         (void) memcpy(ie, ie_in->wpa_ie, ie_in->wpa_ie_len);
1960         if (ic->ic_opt_ie != NULL) {
1961                 ieee80211_dbg(IEEE80211_MSG_BRUSSELS,
1962                     "wl_set_optie: ic_opt_ie != NULL\n");
2099                     "wl_set_optie:ic_opt_ie!=NULL\n");
1963                 ieee80211_free(ic->ic_opt_ie);
1964         }
1965         ic->ic_opt_ie = ie;
1966         ic->ic_opt_ie_len = ie_in->wpa_ie_len;

1968         return (err);
1969 }
______unchanged_portion_omitted_

2000 /*
2001  * MAC_PROP_WL_MLME
2002  */

2004 static int
2005 wl_set_mlme(struct ieee80211com *ic, const void *wldp_buf)
2006 {
2007         int err = 0;
```



```
2008         uint32_t flags;
2009         ieee80211_node_t *in;
2010         wl_mlme_t *mlme = (wl_mlme_t *)wldp_buf;

2012         ieee80211_dbg(IEEE80211_MSG_WPA, "wl_set_mlme: op=%d <%s> \n",
2013             mlme->im_op, ieee80211_macaddr_sprintf(mlme->im_macaddr));
2149         ieee80211_dbg(IEEE80211_MSG_WPA, "wl_set_mlme: "
2150             "op=%d\n", mlme->im_op);

2015         switch (mlme->im_op) {
2016         case IEEE80211_MLME_DISASSOC:
2017         case IEEE80211_MLME_DEAUTH:
2018                 if (ic->ic_opmode == IEEE80211_M_STA) {
2019                         /*
2020                          * Mask ic_flags of IEEE80211_F_WPA to disable
2021                          * ieee80211_notify temporarily.
2022                          */
2023                         flags = ic->ic_flags;
2024                         ic->ic_flags &= ~IEEE80211_F_WPA;

2026                         IEEE80211_UNLOCK(ic);
2027                         ieee80211_new_state(ic, IEEE80211_S_INIT,
2028                             mlme->im_reason);
2029                         IEEE80211_LOCK(ic);

2031                         ic->ic_flags = flags;
2032                 }
2033                 break;
2034         case IEEE80211_MLME_ASSOC:
2035                 if (ic->ic_opmode != IEEE80211_M_STA) {
2036                         ieee80211_err("wifi_cfg_setmlme: opmode err\n");
2037                         err = EINVAL;
2038                         break;
2039                 }
2040                 if (ic->ic_flags & IEEE80211_F_DESBSSID) {
2041                         in = ieee80211_find_node(&ic->ic_scan,
2042                             ic->ic_des_bssid);
2043                 } else if (ic->ic_des_esslen != 0) {
2177                 if (ic->ic_des_esslen != 0) {
2044                 /*
2045                  * Desired ssid specified; must match both bssid and
2046                  * ssid to distinguish ap advertising multiple ssid's.
2047                  */
2048                         in = ieee80211_find_node_with_ssid(&ic->ic_scan,
2049                             mlme->im_macaddr,
2050                             ic->ic_des_esslen,
2051                             ic->ic_des_essid);
2052                 } else {
2053                 /*
2054                  * Normal case; just match bssid.
2055                  */
2056                         in = ieee80211_find_node(&ic->ic_scan,
2057                             mlme->im_macaddr);
2058                 }
2059                 if (in == NULL) {
2060                         ieee80211_err("wifi_cfg_setmlme: "
2061                             "no matched node\n");
2062                         err = EINVAL;
2063                         break;
2064                 }
2065                 IEEE80211_UNLOCK(ic);
2066                 ieee80211_sta_join(ic, in);
2067                 IEEE80211_LOCK(ic);
2068                 break;
2069         default:
2070                 err = EINVAL;
```

```
2071                break;
2072        }

2074        return (err);
2075 }
```

*unchanged_portion_omitted_*

```
2234 /*
2235  * Typically invoked by drivers in response to request for
2236  * information or to change settings from the userland.
2237  *
2238  * Return value should be checked by WiFI drivers. Return 0
2239  * on success. Otherwise, return non-zero value to indicate
2240  * the error. Driver should operate as below when the return
2241  * error is:
2242  * ENETRESET    Reset wireless network and re-start to join a
2243  *              WLAN, ENETRESET is returned when a configuration
2244  *              parameter has been changed.
2245  *              When acknowledge a M_IOCTL message, this error
2246  *              is ignored
2247  */

2249 /*
2250  * Not implemented:
2251  * MAC_PROP_WL_POWER_MODE
2252  * MAC_PROP_WL_RADIO,
2253  */

2255 #endif /* ! codereview */
2256 /* ARGSUSED */
2257 int
2258 ieee80211_setprop(void *ic_arg, const char *pr_name, mac_prop_id_t wldp_pr_num,
2259     uint_t wldp_length, const void *wldp_buf)
2260 {
2261         int err = 0;
2262         struct ieee80211com *ic = ic_arg;

2264         ASSERT(ic != NULL);
2265         IEEE80211_LOCK(ic);

2267         switch (wldp_pr_num) {
2268         /* mac_prop_id */
2269         case MAC_PROP_WL_ESSID:
2270                 err = wl_set_essid(ic, wldp_buf);
2271                 break;
2272         case MAC_PROP_WL_BSSID:
2273                 err = wl_set_bssid(ic, wldp_buf);
2274                 break;
2275         case MAC_PROP_WL_PHY_CONFIG:
2276                 err = wl_set_phy(ic, wldp_buf);
2277                 break;
2278         case MAC_PROP_WL_KEY_TAB:
2279                 err = wl_set_wepkey(ic, wldp_buf);
2280                 break;
2281         case MAC_PROP_WL_AUTH_MODE:
2282                 err = wl_set_authmode(ic, wldp_buf);
2283                 break;
2284         case MAC_PROP_WL_ENCRYPTION:
2285                 err = wl_set_encrypt(ic, wldp_buf);
2286                 break;
2287         case MAC_PROP_WL_BSSTYPE:
2288                 err = wl_set_bsstype(ic, wldp_buf);
2289                 break;
2290         case MAC_PROP_WL_DESIRED_RATES:
2291                 err = wl_set_desrates(ic, wldp_buf);
2292                 break;
```



```
2293         case MAC_PROP_WL_WPA:
2294                 err = wl_set_wpa(ic, wldp_buf);
2295                 break;
2296         case MAC_PROP_WL_KEY:
2297                 err = wl_set_wpakey(ic, wldp_buf);
2298                 break;
2299         case MAC_PROP_WL_DELKEY:
2300                 err = wl_set_delkey(ic, wldp_buf);
2301                 break;
2302         case MAC_PROP_WL_SETOPTIE:
2303                 err = wl_set_optie(ic, wldp_buf);
2304                 break;
2305         case MAC_PROP_WL_MLME:
2306                 err = wl_set_mlme(ic, wldp_buf);
2307                 break;
2308         case MAC_PROP_WL_CREATE_IBSS:
2309                 err = wl_set_createibss(ic, wldp_buf);
2310                 break;
2311         case MAC_PROP_WL_LINKSTATUS:
2312         case MAC_PROP_WL_ESS_LIST:
2313         case MAC_PROP_WL_SUPPORTED_RATES:
2314         case MAC_PROP_WL_RSSI:
2315         case MAC_PROP_WL_CAPABILITY:
2382         case MAC_PROP_WL_SCANRESULTS:
2383                 ieee80211_err("ieee80211_setprop: opmode err\n");
2384                 err = EINVAL;
2385                 break;
2316         default:
2317                 ieee80211_err("ieee80211_setprop: opmode not support\n");
2318                 err = ENOTSUP;
2319                 break;
2320         }

2322         IEEE80211_UNLOCK(ic);

2324         return (err);
2325 }

2327 /* ARGSUSED */
2328 int
2329 ieee80211_getprop(void *ic_arg, const char *pr_name, mac_prop_id_t wldp_pr_num,
2330     uint_t wldp_length, void *wldp_buf)
2331 {
2332         int err = 0;
2333         struct ieee80211com *ic = ic_arg;

2335         ASSERT(ic != NULL);
2336         IEEE80211_LOCK(ic);

2338         switch (wldp_pr_num) {
2339         /* mac_prop_id */
2340         case MAC_PROP_WL_ESSID:
2341                 wl_get_essid(ic, wldp_buf);
2342                 break;
2343         case MAC_PROP_WL_BSSID:
2344                 wl_get_bssid(ic, wldp_buf);
2345                 break;
2346         case MAC_PROP_WL_PHY_CONFIG:
2347                 err = wl_get_phy(ic, wldp_buf);
2348                 break;
2349         case MAC_PROP_WL_AUTH_MODE:
2350                 wl_get_authmode(ic, wldp_buf);
2351                 break;
2352         case MAC_PROP_WL_ENCRYPTION:
2353                 wl_get_encrypt(ic, wldp_buf);
2354                 break;
```

```
2355         case MAC_PROP_WL_BSSTYPE:
2356                 wl_get_bsstype(ic, wldp_buf);
2357                 break;
2358         case MAC_PROP_WL_DESIRED_RATES:
2359                 wl_get_desrates(ic, wldp_buf);
2360                 break;
2361         case MAC_PROP_WL_LINKSTATUS:
2362                 wl_get_linkstatus(ic, wldp_buf);
2363                 break;
2364         case MAC_PROP_WL_ESS_LIST:
2365                 wl_get_esslist(ic, wldp_buf);
2366                 break;
2367         case MAC_PROP_WL_SUPPORTED_RATES:
2368                 wl_get_suprates(ic, wldp_buf);
2369                 break;
2370         case MAC_PROP_WL_RSSI:
2371                 wl_get_rssi(ic, wldp_buf);
2372                 break;
2373         case MAC_PROP_WL_CAPABILITY:
2374                 wl_get_capability(ic, wldp_buf);
2375                 break;
2376         case MAC_PROP_WL_WPA:
2377                 wl_get_wpa(ic, wldp_buf);
2378                 break;
2449         case MAC_PROP_WL_SCANRESULTS:
2450                 wl_get_scanresults(ic, wldp_buf);
2451                 break;
2379         case MAC_PROP_WL_CREATE_IBSS:
2380                 wl_get_createibss(ic, wldp_buf);
2381                 break;
2382         case MAC_PROP_WL_KEY_TAB:
2383         case MAC_PROP_WL_KEY:
2384         case MAC_PROP_WL_DELKEY:
2385         case MAC_PROP_WL_SETOPTIE:
2386         case MAC_PROP_WL_MLME:
2387                 ieee80211_err("ieee80211_setprop: opmode err\n");
2388                 err = EINVAL;
2389                 break;
2390         default:
2391                 ieee80211_err("ieee80211_setprop: opmode not support\n");
2392                 err = ENOTSUP;
2393                 break;
2394         }

2396         IEEE80211_UNLOCK(ic);

2398         return (err);
2399 }

2401 void ieee80211_propinfo(void *ic_arg, const char *pr_name,
2402     mac_prop_id_t wldp_pr_num, mac_prop_info_handle_t prh)
2403 {
2404         _NOTE(ARGUNUSED(pr_name, ic_arg));

2406         /*
2407          * By default permissions are read/write unless specified
2408          * otherwise by the driver.
2409          */

2411         switch (wldp_pr_num) {
2412         case MAC_PROP_WL_LINKSTATUS:
2413         case MAC_PROP_WL_ESS_LIST:
2414         case MAC_PROP_WL_SUPPORTED_RATES:
2415         case MAC_PROP_WL_RSSI:
2416         case MAC_PROP_WL_CAPABILITY:
2490         case MAC_PROP_WL_SCANRESULTS:
```



```
2417         case MAC_PROP_WL_CREATE_IBSS:
2418                 mac_prop_info_set_perm(prh, MAC_PROP_PERM_READ);
2419         }
2420 }
```

_____***unchanged_portion_omitted_***

```
**********************************************************
    1444 Tue Jun 12 19:55:27 2012
new/usr/src/uts/common/net/Makefile
ess_list ioctl now provides all scan results properties for wpa/libdlwlan
first integration of wpa_s control interface client code
first integration of wpa_s wpa_ie parsing code
**********************************************************
```

```
#
# 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.
#
# uts/common/net/Makefile
#
# include global definitions
include ../../../Makefile.master

HDRS=   af.h if.h if_arp.h if_dl.h if_types.h route.h pfkeyv2.h pfpolicy.h \
        ppp-comp.h ppp_defs.h pppio.h vjcompress.h sppptun.h pppoe.h radix.h \
        simnet.h bridge.h bridge_impl.h trill.h
        wpa.h simnet.h bridge.h bridge_impl.h trill.h

ROOTDIRS= $(ROOT)/usr/include/net

ROOTHDRS= $(HDRS:%=$(ROOT)/usr/include/net/%)

CHECKHDRS= $(HDRS:%.h=%.check)

$(ROOTDIRS)/%: %
        $(INS.file)

.KEEP_STATE:

.PARALLEL: $(CHECKHDRS)

install_h: $(ROOTDIRS) $(ROOTHDRS)

$(ROOTDIRS):
        $(INS.dir)

check:  $(CHECKHDRS)
```

```
**********************************************************
   11271 Tue Jun 12 19:55:28 2012
new/usr/src/uts/common/sys/dld.h
secobjs types now are "wep, psk, eap, pin"
dladm_wlan_secmode_t and dladm_secobj_class_t are not related anymore
**********************************************************
______unchanged_portion_omitted_

 120 /*
 121  * Secure objects ioctls
 122  * 1.WEP Static key in 64 hex-digits
 123  * 2.WPA 256-bit preshared key in 64 hex-digits
 124  * 3.User password string for EAP methods (except EAP-TLS) in 64 hex-digits
 125  * 4.Client Private key PKCS#11 keystore PIN
 126 #endif /* ! codereview */
 127  */
 128 typedef enum {
 129         DLD_SECOBJ_CLASS_WEP = 0,
 130         DLD_SECOBJ_CLASS_PSK,
 131         DLD_SECOBJ_CLASS_TLS,
 132         DLD_SECOBJ_CLASS_TTLS,
 133         DLD_SECOBJ_CLASS_PEAP
 122         DLD_SECOBJ_CLASS_WEP = 1,
 123         DLD_SECOBJ_CLASS_WPA
 134 } dld_secobj_class_t;
______unchanged_portion_omitted_
```

```
**********************************************************
   18875 Tue Jun 12 19:55:28 2012
new/usr/src/uts/common/sys/mac.h
Fixed minor compile errors
**********************************************************
______unchanged_portion_omitted_

 136 #define MAXLINKPROPNAME         256             /* max property name len */

 138 /*
 139  * Public properties.
 140  *
 141  * Note that there are 2 sets of parameters: the *_EN_* values are
 142  * those that the Administrator configures for autonegotiation. The
 143  * _ADV_* values are those that are currently exposed over the wire.
 144  */
 145 typedef enum {
 146         MAC_PROP_DUPLEX = 0x00000001,
 147         MAC_PROP_SPEED,
 148         MAC_PROP_STATUS,
 149         MAC_PROP_AUTONEG,
 150         MAC_PROP_EN_AUTONEG,
 151         MAC_PROP_MTU,
 152         MAC_PROP_ZONE,
 153         MAC_PROP_AUTOPUSH,
 154         MAC_PROP_FLOWCTRL,
 155         MAC_PROP_ADV_1000FDX_CAP,
 156         MAC_PROP_EN_1000FDX_CAP,
 157         MAC_PROP_ADV_1000HDX_CAP,
 158         MAC_PROP_EN_1000HDX_CAP,
 159         MAC_PROP_ADV_100FDX_CAP,
 160         MAC_PROP_EN_100FDX_CAP,
 161         MAC_PROP_ADV_100HDX_CAP,
 162         MAC_PROP_EN_100HDX_CAP,
 163         MAC_PROP_ADV_10FDX_CAP,
 164         MAC_PROP_EN_10FDX_CAP,
 165         MAC_PROP_ADV_10HDX_CAP,
 166         MAC_PROP_EN_10HDX_CAP,
 167         MAC_PROP_ADV_100T4_CAP,
 168         MAC_PROP_EN_100T4_CAP,
 169         MAC_PROP_IPTUN_HOPLIMIT,
 170         MAC_PROP_IPTUN_ENCAPLIMIT,
 171         MAC_PROP_WL_ESSID,
 172         MAC_PROP_WL_BSSID,
 173         MAC_PROP_WL_BSSTYPE,
 174         MAC_PROP_WL_LINKSTATUS,
 175         MAC_PROP_WL_DESIRED_RATES,
 176         MAC_PROP_WL_SUPPORTED_RATES,
 177         MAC_PROP_WL_AUTH_MODE,
 178         MAC_PROP_WL_ENCRYPTION,
 179         MAC_PROP_WL_RSSI,
 180         MAC_PROP_WL_PHY_CONFIG,
 181         MAC_PROP_WL_CAPABILITY,
 182         MAC_PROP_WL_WPA,
 183         MAC_PROP_WL_SCANRESULTS,
 183         MAC_PROP_WL_POWER_MODE,
 184         MAC_PROP_WL_RADIO,
 185         MAC_PROP_WL_ESS_LIST,
 186         MAC_PROP_WL_KEY_TAB,
 187         MAC_PROP_WL_CREATE_IBSS,
 188         MAC_PROP_WL_SETOPTIE,
 189         MAC_PROP_WL_DELKEY,
 190         MAC_PROP_WL_KEY,
 191         MAC_PROP_WL_MLME,
 192         MAC_PROP_TAGMODE,
 193         MAC_PROP_ADV_10GFDX_CAP,
```



```
 194         MAC_PROP_EN_10GFDX_CAP,
 195         MAC_PROP_PVID,
 196         MAC_PROP_LLIMIT,
 197         MAC_PROP_LDECAY,
 198         MAC_PROP_RESOURCE,
 199         MAC_PROP_RESOURCE_EFF,
 200         MAC_PROP_RXRINGSRANGE,
 201         MAC_PROP_TXRINGSRANGE,
 202         MAC_PROP_MAX_TX_RINGS_AVAIL,
 203         MAC_PROP_MAX_RX_RINGS_AVAIL,
 204         MAC_PROP_MAX_RXHWCLNT_AVAIL,
 205         MAC_PROP_MAX_TXHWCLNT_AVAIL,
 206         MAC_PROP_IB_LINKMODE,
 207         MAC_PROP_PRIVATE = -1
 208 } mac_prop_id_t;
______unchanged_portion_omitted_
```

```
**********************************************************
   30592 Tue Jun 12 19:55:30 2012
new/usr/src/uts/common/sys/net80211.h
ess_list ioctl now provides all scan results properties for wpa/libdlwlan
first integration of wpa_s control interface client code
first integration of wpa_s wpa_ie parsing code
**********************************************************
```

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

/*
 * Copyright (c) 2001 Atsushi Onoe
 * Copyright (c) 2002-2005 Sam Leffler, Errno Consulting
 * All rights reserved.
 *
 * Redistribution and use in source and binary forms, with or without
 * modification, are permitted provided that the following conditions
 * are met:
 * 1. Redistributions of source code must retain the above copyright
 *    notice, this list of conditions and the following disclaimer.
 * 2. 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.
 * 3. The name of the author may not be used to endorse or promote products
 *    derived from this software without specific prior written permission.
 *
 * Alternatively, this software may be distributed under the terms of the
 * GNU General Public License ("GPL") version 2 as published by the Free
 * Software Foundation.
 *
 * THIS SOFTWARE IS PROVIDED BY THE AUTHOR ''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 AUTHOR 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.
 */

#ifndef _SYS_NET80211_H
#define _SYS_NET80211_H

#include <sys/mac.h>
#include <sys/mac_provider.h>
#include <sys/ethernet.h>
#include <sys/net80211_proto.h>
#include <sys/net80211_crypto.h>
#include <sys/net80211_ht.h>
#include <net/wpa.h>

/*
 * IEEE802.11 kernel support module
 */

#ifdef __cplusplus
extern "C" {
#endif

/* ic_caps */
#define IEEE80211_C_WEP         0x00000001      /* CAPABILITY: WEP available */
#define IEEE80211_C_TKIP        0x00000002      /* CAPABILITY: TKIP available */
```



```
#define IEEE80211_C_AES         0x00000004      /* CAPABILITY: AES OCB avail */
#define IEEE80211_C_AES_CCM     0x00000008      /* CAPABILITY: AES CCM avail */
#define IEEE80211_C_CKIP        0x00000010      /* CAPABILITY: CKIP available */
#define IEEE80211_C_FF          0x00000040      /* CAPABILITY: ATH FF avail */
#define IEEE80211_C_TURBOP      0x00000080
                                /* CAPABILITY: ATH Turbo available */
#define IEEE80211_C_IBSS        0x00000100      /* CAPABILITY: IBSS available */
#define IEEE80211_C_PMGT        0x00000200      /* CAPABILITY: Power mgmt */
#define IEEE80211_C_HOSTAP      0x00000400      /* CAPABILITY: HOSTAP avail */
#define IEEE80211_C_AHDEMO      0x00000800      /* CAPABILITY: Old Adhoc Demo */
#define IEEE80211_C_SWRETRY     0x00001000      /* CAPABILITY: sw tx retry */
#define IEEE80211_C_TXPMGT      0x00002000      /* CAPABILITY: tx power mgmt */
#define IEEE80211_C_SHSLOT      0x00004000      /* CAPABILITY: short slottime */
#define IEEE80211_C_SHPREAMBLE  0x00008000      /* CAPABILITY: short preamble */
#define IEEE80211_C_MONITOR     0x00010000      /* CAPABILITY: monitor mode */
#define IEEE80211_C_TKIPMIC     0x00020000      /* CAPABILITY: TKIP MIC avail */
#define IEEE80211_C_WPA1        0x00800000      /* CAPABILITY: WPA1 avail */
#define IEEE80211_C_WPA2        0x01000000      /* CAPABILITY: WPA2 avail */
#define IEEE80211_C_WPA         0x01800000
                                /* CAPABILITY: WPA1+WPA2 avail */
#define IEEE80211_C_BURST       0x02000000      /* CAPABILITY: frame bursting */
#define IEEE80211_C_WME         0x04000000      /* CAPABILITY: WME avail */
#define IEEE80211_C_WDS         0x08000000      /* CAPABILITY: 4-addr support */
/* 0x10000000 reserved */
#define IEEE80211_C_BGSCAN      0x20000000      /* CAPABILITY: bg scanning */
#define IEEE80211_C_TXFRAG      0x40000000      /* CAPABILITY: tx fragments */
/* XXX protection/barker? */

#define IEEE80211_C_CRYPTO      0x0000001f      /* CAPABILITY: crypto alg's */

/*
 * ic_htcaps: HT-specific device/driver capabilities
 *
 * NB: the low 16-bits are the 802.11 definitions, the upper
 *     16-bits are used to define s/w/driver capabilities.
 */
#define IEEE80211_HTC_AMPDU     0x00010000      /* CAPABILITY: A-MPDU tx */
#define IEEE80211_HTC_AMSDU     0x00020000      /* CAPABILITY: A-MSDU tx */
/* NB: HT40 is implied by IEEE80211_HTCAP_CHWIDTH40 */
#define IEEE80211_HTC_HT        0x00040000      /* CAPABILITY: HT operation */

/* ic_flags */
/* NB: bits 0x4c available */
#define IEEE80211_F_FF          0x00000001      /* CONF: ATH FF enabled */
#define IEEE80211_F_TURBOP      0x00000002      /* CONF: ATH Turbo enabled */
#define IEEE80211_F_BURST       0x00000004      /* CONF: bursting enabled */
/* NB: this is intentionally setup to be IEEE80211_CAPINFO_PRIVACY */
#define IEEE80211_F_PRIVACY     0x00000010      /* CONF: privacy enabled */
#define IEEE80211_F_PUREG       0x00000020      /* CONF: 11g w/o 11b sta's */
#define IEEE80211_F_SCANONLY    0x00000040      /* CONF: scan only */
#define IEEE80211_F_SCAN        0x00000080      /* STATUS: scanning */
#define IEEE80211_F_ASCAN       0x00000100      /* STATUS: active scan */
#define IEEE80211_F_SIBSS       0x00000200      /* STATUS: start IBSS */
/* NB: this is intentionally setup to be IEEE80211_CAPINFO_SHORT_SLOTTIME */
#define IEEE80211_F_SHSLOT      0x00000400
                                /* STATUS: use short slot time */
#define IEEE80211_F_PMGTON      0x00000800      /* CONF: Power mgmt enable */
#define IEEE80211_F_DESBSSID    0x00001000      /* CONF: des_bssid is set */
#define IEEE80211_F_WME         0x00002000      /* CONF: enable WME use */
#define IEEE80211_F_BGSCAN      0x00004000
                                /* CONF: bg scan enabled (???) */
#define IEEE80211_F_SWRETRY     0x00008000      /* CONF: sw tx retry enabled */
#define IEEE80211_F_TXPOW_FIXED 0x00010000      /* TX Power: fixed rate */
#define IEEE80211_F_IBSSON      0x00020000      /* CONF: IBSS creation enable */
#define IEEE80211_F_SHPREAMBLE  0x00040000      /* STATUS: use short preamble */
#define IEEE80211_F_DATAPAD     0x00080000      /* CONF: do alignment pad */
```

```
125 #define IEEE80211_F_USEPROT     0x00100000      /* STATUS: protection enabled */
126 #define IEEE80211_F_USEBARKER   0x00200000
127                                 /* STATUS: use barker preamble */
128 #define IEEE80211_F_TIMUPDATE   0x00400000      /* STATUS: update beacon tim */
129 #define IEEE80211_F_WPA1        0x00800000      /* CONF: WPA enabled */
130 #define IEEE80211_F_WPA2        0x01000000      /* CONF: WPA2 enabled */
131 #define IEEE80211_F_WPA         0x01800000      /* CONF: WPA/WPA2 enabled */
132 #define IEEE80211_F_DROPUNENC   0x02000000      /* CONF: drop unencrypted */
133 #define IEEE80211_F_COUNTERM    0x04000000      /* CONF: TKIP countermeasures */
134 #define IEEE80211_F_HIDESSID    0x08000000      /* CONF: hide SSID in beacon */
135 #define IEEE80211_F_NOBRIDGE    0x10000000      /* CONF: dis. internal bridge */
136 #define IEEE80211_F_WMEUPDATE   0x20000000      /* STATUS: update beacon wme */

138 /* ic_flags_ext */
139 #define IEEE80211_FEXT_NONHT_PR 0x00000001      /* STATUS: non-HT sta present */
140 #define IEEE80211_FEXT_INACT    0x00000002      /* CONF: sta inact handling */
141 /* 0x00000006 reserved */
142 #define IEEE80211_FEXT_BGSCAN   0x00000008
143                                 /* STATUS: enable full bgscan completion */
144 #define IEEE80211_FEXT_ERPUPDATE 0x00000200     /* STATUS: update ERP element */
145 #define IEEE80211_FEXT_SWBMISS  0x00000400      /* CONF: do bmiss in s/w */
146 #define IEEE80211_FEXT_PROBECHAN 0x00020000     /* CONF: probe passive chan */
147 #define IEEE80211_FEXT_HT       0x00080000      /* CONF: HT supported */
148 #define IEEE80211_FEXT_AMPDU_TX 0x00100000      /* CONF: A-MPDU tx supported */
149 #define IEEE80211_FEXT_AMPDU_RX 0x00200000      /* CONF: A-MPDU tx supported */
150 #define IEEE80211_FEXT_AMSDU_TX 0x00400000      /* CONF: A-MSDU tx supported */
151 #define IEEE80211_FEXT_AMSDU_RX 0x00800000      /* CONF: A-MSDU tx supported */
152 #define IEEE80211_FEXT_USEHT40  0x01000000      /* CONF: 20/40 use enabled */
153 #define IEEE80211_FEXT_PUREN    0x02000000      /* CONF: 11n w/o legacy sta's */
154 #define IEEE80211_FEXT_SHORTGI20 0x04000000     /* CONF: short GI in HT20 */
155 #define IEEE80211_FEXT_SHORTGI40 0x08000000     /* CONF: short GI in HT40 */
156 #define IEEE80211_FEXT_HTCOMPAT 0x10000000      /* CONF: HT vendor OUI's */

158 /*
159  * Channel attributes (ich_flags)
160  * bits 0-3 are for private use by drivers
161  */
162 #define IEEE80211_CHAN_TURBO    0x00000010 /* Turbo channel */
163 #define IEEE80211_CHAN_CCK      0x00000020 /* CCK channel */
164 #define IEEE80211_CHAN_OFDM     0x00000040 /* OFDM channel */
165 #define IEEE80211_CHAN_2GHZ     0x00000080 /* 2 GHz spectrum channel. */
166 #define IEEE80211_CHAN_5GHZ     0x00000100 /* 5 GHz spectrum channel */
167 #define IEEE80211_CHAN_PASSIVE  0x00000200 /* Only passive scan allowed */
168 #define IEEE80211_CHAN_DYN      0x00000400 /* Dynamic CCK-OFDM channel */
169 #define IEEE80211_CHAN_GFSK     0x00000800 /* GFSK channel (FHSS PHY) */
170 #define IEEE80211_CHAN_GSM      0x00001000 /* 900 MHz spectrum channel */
171 #define IEEE80211_CHAN_STURBO   0x00002000 /* 11a static turbo channel only */
172 #define IEEE80211_CHAN_HALF     0x00004000 /* Half rate channel */
173 #define IEEE80211_CHAN_QUARTER  0x00008000 /* Quarter rate channel */
174 #define IEEE80211_CHAN_HT20     0x00010000 /* HT 20 channel */
175 #define IEEE80211_CHAN_HT40U    0x00020000 /* HT 40 channel w/ ext above */
176 #define IEEE80211_CHAN_HT40D    0x00040000 /* HT 40 channel w/ ext below */
177 #define IEEE80211_CHAN_DFS      0x00080000 /* DFS required */
178 #define IEEE80211_CHAN_4MSXMIT  0x00100000 /* 4ms limit on frame length */
179 #define IEEE80211_CHAN_NOADHOC  0x00200000 /* adhoc mode not allowed */
180 #define IEEE80211_CHAN_NOHOSTAP 0x00400000 /* hostap mode not allowed */
181 #define IEEE80211_CHAN_11D      0x00800000 /* 802.11d required */

183 #define IEEE80211_CHAN_HT40     (IEEE80211_CHAN_HT40U | IEEE80211_CHAN_HT40D)
184 #define IEEE80211_CHAN_HT       (IEEE80211_CHAN_HT20 | IEEE80211_CHAN_HT40)

186 #define IEEE80211_CHAN_MAX      255
187 #define IEEE80211_CHAN_BYTES    32      /* howmany(IEEE80211_CHAN_MAX, NBBY) */
188 #define IEEE80211_CHAN_ANY      0xffff  /* token for ''any channel'' */
189 #define IEEE80211_CHAN_ANYC     \
190         ((struct ieee80211_channel *)IEEE80211_CHAN_ANY)
```



```
192 #define IEEE80211_IS_CHAN_2GHZ(_c)      \
193         (((_c)->ich_flags & IEEE80211_CHAN_2GHZ) != 0)
194 #define IEEE80211_IS_CHAN_5GHZ(_c)      \
195         (((_c)->ich_flags & IEEE80211_CHAN_5GHZ) != 0)

197 #define IEEE80211_NODE_CHWUPDATE 0x0400         /* 11n channel width change */
198 #define IEEE80211_NODE_HASHSIZE 32

200 #define IEEE80211_NODE_AUTH     0x0001          /* authorized for data */
201 #define IEEE80211_NODE_QOS      0x0002          /* QoS enabled */
202 #define IEEE80211_NODE_ERP      0x0004          /* ERP enabled */
203 /* NB: this must have the same value as IEEE80211_FC1_PWR_MGT */
204 #define IEEE80211_NODE_PWR_MGT  0x0010          /* power save mode enabled */
205 #define IEEE80211_NODE_AREF     0x0020          /* authentication ref held */
206 #define IEEE80211_NODE_HT       0x0040          /* HT enabled */
207 #define IEEE80211_NODE_HTCOMPAT 0x0080          /* HT setup w/ vendor OUI's */
208 #define IEEE80211_NODE_AMPDU_RX 0x0400          /* AMPDU rx enabled */
209 #define IEEE80211_NODE_AMPDU_TX 0x0800          /* AMPDU tx enabled */

211 #define IEEE80211_NODE_AMPDU \
212         (IEEE80211_NODE_AMPDU_RX | IEEE80211_NODE_AMPDU_TX)

214 #define IEEE80211_FIXED_RATE_NONE       0

216 #define WME_OUI                 0xf25000
217 #define WME_OUI_TYPE            0x02
218 #define WME_INFO_OUI_SUBTYPE    0x00
219 #define WME_PARAM_OUI_SUBTYPE   0x01
220 #define WME_VERSION             1

222 /* WME stream classes */
223 #define WME_AC_BE               0       /* best effort */
224 #define WME_AC_BK               1       /* background */
225 #define WME_AC_VI               2       /* video */
226 #define WME_AC_VO               3       /* voice */

228 #define MAX_EVENT               16
229 #define MAX_IEEE80211STR        256

231 /* required for 'arn' driver */
232 #define MAX_RSSI                15

234 #endif /* ! codereview */
235 /* For IEEE80211_RADIOTAP_FLAGS */
236 #define IEEE80211_RADIOTAP_F_CFP        0x01
237                                         /* sent/received during CFP */
238 #define IEEE80211_RADIOTAP_F_SHORTPRE   0x02
239                                         /* sent/received with short preamble */
240 #define IEEE80211_RADIOTAP_F_WEP        0x04
241                                         /* sent/received with WEP encryption */
242 #define IEEE80211_RADIOTAP_F_FRAG       0x08
243                                         /* sent/received with fragmentation */
244 #define IEEE80211_RADIOTAP_F_DATAPAD    0x20
245                                         /*
246                                          * frame has padding between 802.11
247                                          * header and payload (to 32-bit
248                                          * boundary
249                                          */
250 #define IEEE80211_RADIOTAP_F_FCS        0x10    /* frame includes FCS */
251 #define IEEE80211_RADIOTAP_F_BADFCS     0x40    /* does not pass FCS check */
252 #define IEEE80211_RADIOTAP_F_SHORTGI    0x80    /* HT short GI */

254 /*
255  * Authentication mode.
256  * auth_algs should be only OPEN/SHARED/LEAP
```

```
 * see IEEE80211_AUTH_ALG_* in net80211_proto.h
 * NONE/8021X/AUTO/WPA refer to key_mgmt protocols!
#endif /* ! codereview */
 */
enum ieee80211_authmode {
        IEEE80211_AUTH_NONE     = 0,
        IEEE80211_AUTH_OPEN     = 1,    /* open */
        IEEE80211_AUTH_SHARED   = 2,    /* shared-key */
        IEEE80211_AUTH_8021X    = 3,    /* 802.1x */
        IEEE80211_AUTH_AUTO     = 4,    /* auto-select/accept */
        /* actually, this is never used */
        /* NB: these are used only for ioctls */
        IEEE80211_AUTH_WPA      = 5     /* WPA/RSN w/ 802.1x/PSK */
};
```

*unchanged_portion_omitted_*

```
#endif /* ! codereview */
#define IEEE80211_S_MAX (IEEE80211_S_RUN+1)

#define WPA_DOOR        "/var/run/wpa_door"

/*
 * WPA State machine events
 */
typedef enum {
        EVENT_ASSOC,
        EVENT_DISASSOC,
        EVENT_SCAN_RESULTS = 3
} wpa_event_type;

typedef struct  wl_events {
        wpa_event_type  event;
} wl_events_t;


#endif /* ! codereview */
/*
 * 802.11 rate set.
 */
#define IEEE80211_RATE_MAXSIZE  15      /* max rates we'll handle */
#define IEEE80211_RATE_SIZE     8       /* 802.11 standard */
#define IEEE80211_XRATE_SIZE    (IEEE80211_RATE_MAXSIZE - IEEE80211_RATE_SIZE)
                                        /* size of extended supported rates */
struct ieee80211_rateset {
        uint8_t                 ir_nrates;
        uint8_t                 ir_rates[IEEE80211_RATE_MAXSIZE];
};

/*
 * 802.11n variant of ieee80211_rateset.  Instead
 * legacy rates the entries are MCS rates.  We define
 * the structure such that it can be used interchangeably
 * with an ieee80211_rateset (modulo structure size).
 */
#define IEEE80211_HTRATE_MAXSIZE 127

struct ieee80211_htrateset {
        uint8_t                 rs_nrates;
        uint8_t                 rs_rates[IEEE80211_HTRATE_MAXSIZE];
};

#define IEEE80211_RATE_MCS      0x80

/*
 * Channels are specified by frequency and attributes.
 */
```



```
struct ieee80211_channel {
        uint16_t                ich_freq;       /* setting in Mhz */
        uint32_t                ich_flags;      /* see below */
};

struct ieee80211_device_stats {
        uint32_t                is_tx_frags;
        uint32_t                is_tx_bytes;
        uint32_t                is_tx_mcast;
        uint32_t                is_tx_failed;
        uint32_t                is_tx_retries;
        uint32_t                is_rts_success;
        uint32_t                is_rts_failure;
        uint32_t                is_ack_failure;
        uint32_t                is_rx_frags;
        uint32_t                is_rx_bytes;
        uint32_t                is_rx_mcast;
        uint32_t                is_rx_dups;
        uint32_t                is_fcs_errors;
        uint32_t                is_wep_errors;
        uint32_t                is_tx_nobuf;
        uint32_t                is_tx_unknownmgt;
};

struct ieee80211_crypto_state;
typedef struct ieee80211_node_table ieee80211_node_table_t;
typedef struct ieee80211_node ieee80211_node_t;
typedef struct ieee80211com ieee80211com_t;

struct ieee80211_node_table {
        struct ieee80211com     *nt_ic;         /* back reference */

        const char              *nt_name;       /* for debugging */
        /* For node inactivity processing */
        int                     nt_inact_timer; /* inactivity timer */
        int                     nt_inact_init;  /* initial node inact setting */
        void                    (*nt_timeout)(struct ieee80211_node_table *);
        uint32_t                nt_scangen;     /* gen# for timeout scan */
        kmutex_t                nt_scanlock;    /* on nt_scangen */
        kmutex_t                nt_nodelock;    /* on node table */

        int                     nt_keyixmax;    /* keyixmap size */
        struct ieee80211_node   **nt_keyixmap;  /* key ix -> node map */

        list_t                  nt_node;        /* information of all nodes */
        list_t                  nt_hash[IEEE80211_NODE_HASHSIZE];
};

#define IEEE80211_TID_SIZE      (WME_NUM_TID+1) /* WME TID's +1 for non-QoS */
#define IEEE80211_NONQOS_TID    WME_NUM_TID     /* index for non-QoS sta */

/*
 * Node specific information.  Note that drivers are expected
 * to derive from this structure to add device-specific per-node
 * state.  This is done by overriding the ic_node_* methods in
 * the ieee80211com structure.
 */
struct ieee80211_node {
        struct ieee80211com             *in_ic;
        struct ieee80211_node_table     *in_table;

        uint8_t                 in_authmode;    /* authentication algorithm */
        uint16_t                in_flags;       /* special purpose state */
        uint16_t                in_associd;     /* assoc response */
        uint16_t                in_txpower;     /* current transmit power */
        uint16_t                in_vlan;        /* vlan tag */
```

```
395         /*
396          * Tx/Rx sequence number.
397          * index 0 is used when QoS is not enabled. index 1-16 is used
398          * when QoS is enabled. 1-16 corresponds to TID 0-15.
399          */
400         uint16_t                in_txseqs[IEEE80211_TID_SIZE];
401         uint16_t                in_rxseqs[IEEE80211_TID_SIZE];
402         clock_t                 in_rxfragstamp; /* time stamp of last rx frag */
403         mblk_t                  *in_rxfrag;     /* rx frag reassembly */
404         uint32_t                in_scangen;     /* gen# for timeout scan */
405         uint32_t                in_refcnt;

407         /* hardware */
408         uint32_t                in_rstamp;      /* recv timestamp */
409         uint8_t                 in_rssi;        /* recv ssi */

411         /* header */
412         uint8_t                 in_macaddr[IEEE80211_ADDR_LEN];
413         uint8_t                 in_bssid[IEEE80211_ADDR_LEN];

415         /* beacon, probe response */
416         union {
417                 uint8_t         data[8];
418                 uint64_t        tsf;
419         } in_tstamp;                            /* from last rcv'd beacon */
420         uint16_t                in_intval;      /* beacon interval */
421         uint16_t                in_capinfo;     /* capabilities */
422         uint8_t                 in_esslen;
423         uint8_t                 in_essid[IEEE80211_NWID_LEN];
424         struct ieee80211_rateset in_rates;      /* negotiated rate set */
425         struct ieee80211_channel *in_chan;      /* XXX multiple uses */
426         enum ieee80211_phytype  in_phytype;
427         uint16_t                in_fhdwell;     /* FH only */
428         uint8_t                 in_fhindex;     /* FH only */
429         uint8_t                 in_erp;         /* ERP from beacon/probe resp */
430         uint16_t                in_tim_off;     /* byte offset to TIM ie */
431         uint8_t                 in_dtim_period; /* DTIM period */
432         uint8_t                 in_dtim_count;  /* DTIM count for last bcn */

434         uint32_t                *in_challenge;  /* shared-key challenge */
435         struct ieee80211_key    in_ucastkey;    /* unicast key */
436         uint8_t                 *in_wpa_ie;     /* captured WPA/RSN ie */
437         uint8_t                 *in_wme_ie;     /* captured WME ie */

439         /* 11n state */
440         uint8_t                 *in_htcap_ie;   /* captured HTCAP ie */
441         uint16_t                in_htcap;       /* HT capabilities */
442         uint8_t                 in_htparam;     /* HT params */
443         uint8_t                 in_htctlchan;   /* HT control channel */
444         uint8_t                 in_ht2ndchan;   /* HT 2nd channel */
445         uint8_t                 in_htopmode;    /* HT operating mode */
446         uint8_t                 in_htstbc;      /* HT */
447         uint8_t                 in_reqcw;       /* requested tx channel width */
448         uint8_t                 in_chw;         /* negotiated channel width */
449         struct ieee80211_htrateset in_htrates;  /* negotiated ht rate set */
450         struct ieee80211_tx_ampdu in_tx_ampdu[WME_NUM_AC];
451         struct ieee80211_rx_ampdu in_rx_ampdu[WME_NUM_TID];

453         /* others */
454         int32_t                 in_fails;       /* failure count to associate */
455         int16_t                 in_inact;       /* inactivity mark count */
456         int16_t                 in_inact_reload; /* inactivity reload value */
457         int32_t                 in_txrate;      /* index to in_rates[] */

459         list_node_t             in_node;        /* element of nt->nt_node */
460         list_node_t             in_hash;        /* element of nt->nt_hash */
```



```
461 };

463 /*
464  * WME/WMM support.
465  */
466 struct wmeParams {
467         uint8_t         wmep_acm;
468         uint8_t         wmep_aifsn;
469         uint8_t         wmep_logcwmin;          /* log2(cwmin) */
470         uint8_t         wmep_logcwmax;          /* log2(cwmax) */
471         uint8_t         wmep_txopLimit;
472         uint8_t         wmep_noackPolicy;       /* 0 (ack), 1 (no ack) */
473 };
474 #define IEEE80211_TXOP_TO_US(_txop)     ((_txop)<<5)
475 #define IEEE80211_US_TO_TXOP(_us)       ((_us)>>5)

477 struct chanAccParams {
478         uint8_t         cap_info;               /* version of the current set */
479         struct wmeParams cap_wmeParams[WME_NUM_AC];
480 };

482 struct ieee80211_wme_state {
483         uint_t  wme_flags;
484 #define WME_F_AGGRMODE  0x00000001      /* STATUS: WME agressive mode */
485         uint_t  wme_hipri_traffic; /* VI/VO frames in beacon interval */
486         uint_t  wme_hipri_switch_thresh; /* agressive mode switch thresh */
487         uint_t  wme_hipri_switch_hysteresis;
488                                         /* agressive mode switch hysteresis */
489         struct wmeParams wme_params[4]; /* from assoc resp for each AC */
490         struct chanAccParams wme_wmeChanParams; /* WME params applied to self */
491         struct chanAccParams wme_wmeBssChanParams;
492                                         /* WME params bcast to stations */
493         struct chanAccParams wme_chanParams; /* params applied to self */
494         struct chanAccParams wme_bssChanParams; /* params bcast to stations */
495         int (*wme_update)(struct ieee80211com *);
496 };

498 struct ieee80211com {
499         mac_handle_t            ic_mach;

501         /* Initialized by driver */
502         uint8_t                 ic_macaddr[IEEE80211_ADDR_LEN];
503         uint32_t                ic_caps;        /* capabilities */
504         uint32_t                ic_htcaps;      /* HT capabilities */
505         enum ieee80211_phytype  ic_phytype;     /* XXX wrong for multi-mode */
506         enum ieee80211_opmode   ic_opmode;      /* current operation mode */
507         enum ieee80211_state    ic_state;       /* current 802.11 state */
508         struct ieee80211_channel        ic_sup_channels[IEEE80211_CHAN_MAX+1];
509         struct ieee80211_rateset        ic_sup_rates[IEEE80211_MODE_MAX];
510         enum ieee80211_phymode          ic_curmode;  /* OPT current mode */
511         struct ieee80211_channel        *ic_curchan; /* OPT current channel */
512         struct ieee80211_channel        *ic_ibss_chan;  /* OPT bss channel */
513         uint8_t                         ic_maxrssi;  /* maximum hardware RSSI */

515         /* INITIALIZED by IEEE80211, used/overridden by driver */
516         uint16_t                ic_modecaps;    /* set of mode capabilities */
517         uint8_t                 ic_chan_active[IEEE80211_CHAN_BYTES];
518         enum ieee80211_protmode ic_protmode;    /* 802.11g protection mode */
519         uint16_t                ic_bintval;     /* beacon interval */
520         uint16_t                ic_lintval;     /* listen interval */
521         uint16_t                ic_txpowlimit;  /* global tx power limit */
522         uint8_t                 ic_bmissthreshold;
523         uint16_t                ic_rtsthreshold;
524         uint16_t                ic_fragthreshold;
525         uint8_t                 ic_fixed_rate;  /* value of fixed rate */
526         int32_t                 ic_des_esslen;  /* length of desired essid */
```

```
527         uint8_t                 ic_des_essid[IEEE80211_NWID_LEN];
528         uint8_t                 ic_des_bssid[IEEE80211_ADDR_LEN];
529         struct ieee80211_channel        *ic_des_chan;   /* desired channel */
530         void                    *ic_opt_ie;     /* user-specified IE's */
531         uint16_t                ic_opt_ie_len;  /* length of ic_opt_ie */
532         uint8_t                 ic_nickname[IEEE80211_NWID_LEN];
533         uint16_t                ic_tim_len;     /* ic_tim_bitmap size (bytes) */
534         uint8_t                 *ic_tim_bitmap; /* powersave stations w/ data */
535         timeout_id_t            ic_watchdog_timer;      /* watchdog timer */
536         /* Cipher state/configuration. */
537         struct ieee80211_crypto_state   ic_crypto;
538         const struct ieee80211_cipher *ic_ciphers[IEEE80211_CIPHER_MAX];

540         kmutex_t                ic_doorlock;
541         char                    ic_wpadoor[MAX_IEEE80211STR];

543         wpa_event_type          ic_eventq[MAX_EVENT];
544         uint32_t                ic_evq_head, ic_evq_tail;

546         /* Runtime states */
547         uint32_t                ic_flags;       /* state/conf flags */
548         uint32_t                ic_flags_ext;   /* extended state flags */
549         struct ieee80211_node   *ic_bss;        /* information for this node */
550         struct ieee80211_device_stats   ic_stats;
551         struct ieee80211_node_table     ic_scan; /* STA: scan candidates */
552         struct ieee80211_node_table     ic_sta; /* AP:stations/IBSS:neighbors */

554         struct ieee80211_wme_state ic_wme;      /* WME/WMM state */

556         int                     ic_ampdu_rxmax; /* A-MPDU rx limit (bytes) */
557         int                     ic_ampdu_density; /* A-MPDU density */
558         int                     ic_ampdu_limit; /* A-MPDU tx limit (bytes) */
559         int                     ic_amsdu_limit; /* A-MSDU tx limit (bytes) */

561         uint16_t                ic_sta_assoc;   /* stations associated */
562         uint16_t                ic_ht_sta_assoc; /* HT stations associated */
563         uint16_t                ic_ht40_sta_assoc; /* HT40 station associated */
564         uint8_t                 ic_curhtprotmode; /* HTINFO bss state */
565         enum ieee80211_protmode ic_htprotmode;  /* HT protection mode */
566         int                     ic_lastnonerp;  /* last time nonERP sta noted */
567         int                     ic_lastnonht;   /* last time non-HT sta noted */
568         int                     ic_beaconmiss;  /* beacon miss counter */


571         /* callback functions */
572         /*
573          * Functions initialized by driver before calling ieee80211_attach()
574          * Those must be initialized are marked with M(andatory)
575          *
576          *  ic_xmit             - [M] transmit a management or null data frame
577          *                      return 0 on success, non-zero on error
578          *  ic_watchdog         - [O] periodic run function, enabled by
579          *                      ieee80211_start_watchdog()
580          *  ic_set_tim          - [O] set/clear traffic indication map
581          *  ic_set_shortslot    - [O] enable/disable short slot timing
582          *  ic_node_newassoc    - [O] driver specific operation on a newly
583          *                      associated or re-assoced node
584          */
585         int                     (*ic_xmit)(ieee80211com_t *, mblk_t *, uint8_t);
586         void                    (*ic_watchdog)(void *);
587         void                    (*ic_set_tim)(ieee80211com_t *,
588                                         ieee80211_node_t *, int);
589         void                    (*ic_set_shortslot)(ieee80211com_t *, int);
590         void                    (*ic_node_newassoc)(ieee80211_node_t *, int);
591         /*
592          * Functions initialized by ieee80211_attach(), driver could
```



```
593          * override these functions after calling ieee80211_attach()
594          *
595          *  ic_reset            - reset
596          *  ic_recv_mgmt        - handle received management frames
597          *  ic_send_mgmt        - construct and transmit management frames
598          *  ic_newstate         - handle state transition
599          *  ic_node_alloc       - allocate a new BSS info node
600          *  ic_node_cleanup     - cleanup or free memory spaces of a node
601          *  ic_node_free        - free a node
602          *  ic_node_getrssi     - get node's rssi
603          */
604         int                     (*ic_reset)(ieee80211com_t *);
605         void                    (*ic_recv_mgmt)(ieee80211com_t *,
606                                         mblk_t *, ieee80211_node_t *,
607                                         int, int, uint32_t);
608         int                     (*ic_send_mgmt)(ieee80211com_t *,
609                                         ieee80211_node_t *, int, int);
610         int                     (*ic_newstate)(ieee80211com_t *,
611                                         enum ieee80211_state, int);
612         struct ieee80211_node   *(*ic_node_alloc)(ieee80211com_t *);
613         void                    (*ic_node_cleanup)(ieee80211_node_t *);
614         void                    (*ic_node_free)(ieee80211_node_t *);
615         uint8_t                 (*ic_node_getrssi)(const ieee80211_node_t *);
616         void                    (*ic_set_channel)(ieee80211com_t *);

618         /*
619          * 802.11n ADDBA support.  A simple/generic implementation
620          * of A-MPDU tx aggregation is provided; the driver may
621          * override these methods to provide their own support.
622          * A-MPDU rx re-ordering happens automatically if the
623          * driver passes out-of-order frames to ieee80211_input
624          * from an assocated HT station.
625          */
626         void                    (*ic_recv_action)(ieee80211_node_t *,
627                                     const uint8_t *, const uint8_t *);
628         int                     (*ic_send_action)(ieee80211_node_t *,
629                                         int, int, uint16_t[4]);
630         /* start/stop doing A-MPDU tx aggregation for a station */
631         int                     (*ic_addba_request)(ieee80211_node_t *,
632                                     struct ieee80211_tx_ampdu *,
633                                     int, int, int);
634         int                     (*ic_addba_response)(ieee80211_node_t *,
635                                     struct ieee80211_tx_ampdu *,
636                                     int, int, int);
637         void                    (*ic_addba_stop)(ieee80211_node_t *,
638                                     struct ieee80211_tx_ampdu *);

640         kmutex_t                ic_genlock;
641         void                    *ic_private;    /* ieee80211 private data */
642 };
643 #define ic_nw_keys              ic_crypto.cs_nw_keys
644 #define ic_def_txkey            ic_crypto.cs_def_txkey

646 extern  const char *ieee80211_state_name[IEEE80211_S_MAX];
647 extern  const char *ieee80211_wme_acnames[];

649 #define IEEE80211_RATE(_ix)                     \
650         (in->in_rates.ir_rates[(_ix)] & IEEE80211_RATE_VAL)

652 #define ieee80211_new_state(_ic, _nstate, _arg) \
653         (((_ic)->ic_newstate)((_ic), (_nstate), (_arg)))

655 #define ieee80211_macaddr_sprintf(_addr)        \
656         ether_sprintf((struct ether_addr *)(_addr))

658 /*
```

```
 * Node reference counting definitions.
 *
 * ieee80211_node_initref       initialize the reference count to 1
 * ieee80211_node_incref        add a reference
 * ieee80211_node_decref        remove a reference
 * ieee80211_node_decref_nv     remove a reference and return new value
 * ieee80211_node_refcnt        reference count for printing (only)
 */
#include <sys/atomic.h>
#define ieee80211_node_initref(_in)             \
        ((_in)->in_refcnt = 1)
#define ieee80211_node_incref(_in)              \
        atomic_inc_uint(&(_in)->in_refcnt)
#define ieee80211_node_decref(_in)              \
        atomic_dec_uint(&(_in)->in_refcnt)
#define ieee80211_node_decref_nv(_in)           \
        atomic_dec_uint_nv(&(_in)->in_refcnt)
#define ieee80211_node_refcnt(_in)              \
        (_in)->in_refcnt

typedef void ieee80211_iter_func(void *, ieee80211_node_t *);

/* Initialization */
void ieee80211_attach(ieee80211com_t *);
void ieee80211_detach(ieee80211com_t *);
void ieee80211_media_init(ieee80211com_t *);
int ieee80211_ioctl(ieee80211com_t *, queue_t *, mblk_t *);
void ieee80211_register_door(ieee80211com_t *, const char *, int);

/* Protocol Processing */
int ieee80211_input(ieee80211com_t *, mblk_t *, ieee80211_node_t *,
        int32_t, uint32_t);
mblk_t *ieee80211_encap(ieee80211com_t *, mblk_t *, ieee80211_node_t *);

mblk_t *ieee80211_beacon_alloc(ieee80211com_t *, ieee80211_node_t *,
        struct ieee80211_beacon_offsets *);
int ieee80211_beacon_update(ieee80211com_t *, ieee80211_node_t *,
        struct ieee80211_beacon_offsets *, mblk_t *, int);
void ieee80211_beacon_miss(ieee80211com_t *);

void ieee80211_begin_scan(ieee80211com_t *, boolean_t);
void ieee80211_next_scan(ieee80211com_t *);
void ieee80211_end_scan(ieee80211com_t *);
void ieee80211_cancel_scan(ieee80211com_t *);

void ieee80211_sta_join(ieee80211com_t *, ieee80211_node_t *);
void ieee80211_sta_leave(ieee80211com_t *, ieee80211_node_t *);
boolean_t ieee80211_ibss_merge(ieee80211_node_t *);

/* Node Operation */
ieee80211_node_t *ieee80211_ref_node(ieee80211_node_t *);
void ieee80211_unref_node(ieee80211_node_t **);
void ieee80211_node_authorize(ieee80211_node_t *);
void ieee80211_node_unauthorize(ieee80211_node_t *);
ieee80211_node_t *ieee80211_alloc_node(ieee80211com_t *,
        ieee80211_node_table_t *, const uint8_t *);
void ieee80211_free_node(ieee80211_node_t *);
void ieee80211_node_table_reset(ieee80211_node_table_t *);
void ieee80211_iterate_nodes(ieee80211_node_table_t *, ieee80211_iter_func *,
        void *);
ieee80211_node_t *ieee80211_find_node(ieee80211_node_table_t *,
        const uint8_t *);
ieee80211_node_t *ieee80211_find_node_with_ssid(ieee80211_node_table_t *,
        const uint8_t *, uint32_t, const uint8_t *);
ieee80211_node_t *ieee80211_find_txnode(ieee80211com_t *,
        const uint8_t daddr[IEEE80211_ADDR_LEN]);
```



```
ieee80211_node_t *ieee80211_find_rxnode(ieee80211com_t *,
        const struct ieee80211_frame *);


/* Crypto */
extern struct ieee80211_key *ieee80211_crypto_encap(ieee80211com_t *, mblk_t *);
extern struct ieee80211_key *ieee80211_crypto_decap(ieee80211com_t *, mblk_t *,
        int);
extern int ieee80211_crypto_newkey(ieee80211com_t *, int, int,
        struct ieee80211_key *);
extern int ieee80211_crypto_delkey(ieee80211com_t *, struct ieee80211_key *);
extern int ieee80211_crypto_setkey(ieee80211com_t *, struct ieee80211_key *,
        const uint8_t macaddr[IEEE80211_ADDR_LEN]);

/* Helper Functions */
int ieee80211_stat(ieee80211com_t *ic, uint_t stat, uint64_t *val);
uint32_t ieee80211_chan2ieee(ieee80211com_t *, struct ieee80211_channel *);
enum ieee80211_phymode ieee80211_chan2mode(ieee80211com_t *,
        struct ieee80211_channel *);
uint32_t ieee80211_ieee2mhz(uint32_t, uint32_t);
void ieee80211_reset_chan(ieee80211com_t *);
void ieee80211_dump_pkt(const uint8_t *, int32_t, int32_t, int32_t);
void ieee80211_watchdog(void *);
void ieee80211_start_watchdog(ieee80211com_t *, uint32_t);
void ieee80211_stop_watchdog(ieee80211com_t *);
int ieee80211_classify(struct ieee80211com *, mblk_t *,
    struct ieee80211_node *);
int ieee80211_hdrsize(const void *);
int ieee80211_hdrspace(ieee80211com_t *, const void *);
int ieee80211_anyhdrsize(const void *);
int ieee80211_anyhdrspace(ieee80211com_t *, const void *);

void *ieee80211_malloc(size_t);
void ieee80211_free(void *);
int ieee80211_setprop(void *, const char *, mac_prop_id_t, uint_t,
    const void *);
int ieee80211_getprop(void *, const char *, mac_prop_id_t, uint_t, void *);
void ieee80211_propinfo(void *, const char *, mac_prop_id_t,
    mac_prop_info_handle_t);


struct ieee80211_channel *ieee80211_find_channel(ieee80211com_t *, int, int);
const struct ieee80211_rateset *ieee80211_get_suprates(ieee80211com_t *,
    struct ieee80211_channel *);

/* HT */

#ifdef  __cplusplus
}
#endif

#endif  /* _SYS_NET80211_H */
```

```
**********************************************************
   32487 Tue Jun 12 19:55:31 2012
new/usr/src/uts/common/sys/net80211_proto.h
ess_list ioctl now provides all scan results properties for wpa/libdlwlan
first integration of wpa_s control interface client code
first integration of wpa_s wpa_ie parsing code
**********************************************************
______unchanged_portion_omitted_

 786 #define BCM_OUI                 0x4c9000        /* Broadcom OUI */
 787 #define BCM_OUI_HTCAP           51              /* pre-draft HTCAP ie */
 788 #define BCM_OUI_HTINFO          52              /* pre-draft HTINFO ie */

 790 #define WPA_OUI                 0xf25000
 791 #define WPA_OUI_TYPE            0x01
 792 //#define       WPA_VERSION             1               /* current supported ver
 792 #define WPA_VERSION             1               /* current supported version */

 794 #define IEEE80211_CHALLENGE_LEN                 128

 796 #define IEEE80211_RATE_BASIC                    0x80
 797 #define IEEE80211_RATE_VAL                      0x7f

 799 /* EPR information element flags */
 800 #define IEEE80211_ERP_NON_ERP_PRESENT           0x01
 801 #define IEEE80211_ERP_USE_PROTECTION            0x02
 802 #define IEEE80211_ERP_LONG_PREAMBLE             0x04

 804 #define IEEE80211_AUTH_ALG_OPEN                 0x0000
 805 #define IEEE80211_AUTH_ALG_SHARED               0x0001
 806 #define IEEE80211_AUTH_ALG_LEAP                 0x0080


 809 enum {
 810         IEEE80211_AUTH_OPEN_REQUEST             = 1,
 811         IEEE80211_AUTH_OPEN_RESPONSE            = 2
 812 };
______unchanged_portion_omitted_
```