```
**********************************************************
    1418 Fri Mar 31 14:29:37 2017
new/usr/src/cmd/smbsrv/dtrace/Makefile
NEX-1643 dtrace provider for smbsrv
Also illumos 1841:
 DTrace smb provider was mis-implemented, doesn't exist.
Add back handlers for read/write raw, so that
 legacy dtrace consumers can find the probes.
Kill extra arg in smb_negotiate
Fix missing "done" probe with smb_notify
Add example consumer: smb-trace.d
fix soi_pid
**********************************************************
```

```
#
# 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 2008 Sun Microsystems, Inc.  All rights reserved.
# Use is subject to license terms.
#
# Copyright 2017 Nexenta Systems, Inc.  All rights reserved.
# Copyright 2014 Nexenta Systems, Inc.  All rights reserved.
#

SRCS=   smbd-all.d smbd-authsvc.d smbd-doorsvc.d smbd-pipesvc.d \
        smbnode.d smbsrv.d smbvfs.d smb-trace.d
        smbnode.d smbsrv.d smbvfs.d

include ../../Makefile.cmd

ROOTSMBDTRACEDIR = $(ROOTLIB)/smbsrv/dtrace
ROOTSMBDTRACEFILE = $(SRCS:%=$(ROOTSMBDTRACEDIR)/%)

$(ROOTSMBDTRACEFILE):= FILEMODE = 0555

$(ROOTSMBDTRACEDIR):
        $(INS.dir)

$(ROOTSMBDTRACEDIR)/%: %
        $(INS.file)

all:

clean:

lint:

include ../../Makefile.targ
```



```
install: all $(ROOTSMBDTRACEDIR) .WAIT $(ROOTSMBDTRACEFILE)
```

```
**********************************************************
    4369 Fri Mar 31 14:29:37 2017
new/usr/src/cmd/smbsrv/dtrace/smb-trace.d
NEX-1643 dtrace provider for smbsrv
Also illumos 1841:
 DTrace smb provider was mis-implemented, doesn't exist.
Add back handlers for read/write raw, so that
 legacy dtrace consumers can find the probes.
Kill extra arg in smb_negotiate
Fix missing "done" probe with smb_notify
Add example consumer: smb-trace.d
fix soi_pid
**********************************************************
```

```
/*
 * This file and its contents are supplied under the terms of the
 * Common Development and Distribution License ("CDDL"), version 1.0.
 * You may only use this file in accordance with the terms of version
 * 1.0 of the CDDL.
 *
 * A full copy of the text of the CDDL should have accompanied this
 * source.  A copy of the CDDL is also available via the Internet at
 * http://www.illumos.org/license/CDDL.
 */

/*
 * Copyright 2017 Nexenta Systems, Inc.  All rights reserved.
 */

/*
 * Example using the "smb" dtrace provider.
 * Traces all SMB commands.
 *
 * All these probes provide:
 *      args[0]  conninfo_t
 *      args[1]  smbopinfo_t
 * Some also provide one of: (not used here)
 *      args[2]  smb_name_args_t
 *      args[2]  smb_open_args_t
 *      args[2]  smb_rw_args_t
 */


smb:::op-CheckDirectory-start,
smb:::op-CloseAndTreeDisconnect-start,
smb:::op-ClosePrintFile-start,
smb:::op-Close-start,
smb:::op-CreateDirectory-start,
smb:::op-CreateNew-start,
smb:::op-CreateTemporary-start,
smb:::op-Create-start,
smb:::op-DeleteDirectory-start,
smb:::op-Delete-start,
smb:::op-Echo-start,
smb:::op-FindClose2-start,
smb:::op-FindClose-start,
smb:::op-FindUnique-start,
smb:::op-Find-start,
smb:::op-Flush-start,
smb:::op-GetPrintQueue-start,
smb:::op-Invalid-start,
smb:::op-Ioctl-start,
smb:::op-LockAndRead-start,
smb:::op-LockByteRange-start,
smb:::op-LockingX-start,
smb:::op-LogoffX-start,
smb:::op-Negotiate-start,
```



```
smb:::op-NtCancel-start,
smb:::op-NtCreateX-start,
smb:::op-NtRename-start,
smb:::op-NtTransactCreate-start,
smb:::op-NtTransactSecondary-start,
smb:::op-NtTransact-start,
smb:::op-OpenPrintFile-start,
smb:::op-OpenX-start,
smb:::op-Open-start,
smb:::op-ProcessExit-start,
smb:::op-QueryInformation2-start,
smb:::op-QueryInformationDisk-start,
smb:::op-QueryInformation-start,
smb:::op-ReadRaw-start,
smb:::op-ReadX-start,
smb:::op-Read-start,
smb:::op-Rename-start,
smb:::op-Search-start,
smb:::op-Seek-start,
smb:::op-SessionSetupX-start,
smb:::op-SetInformation2-start,
smb:::op-SetInformation-start,
smb:::op-Transaction2Secondary-start,
smb:::op-Transaction2-start,
smb:::op-TransactionSecondary-start,
smb:::op-Transaction-start,
smb:::op-TreeConnectX-start,
smb:::op-TreeConnect-start,
smb:::op-TreeDisconnect-start,
smb:::op-UnlockByteRange-start,
smb:::op-WriteAndClose-start,
smb:::op-WriteAndUnlock-start,
smb:::op-WritePrintFile-start,
smb:::op-WriteRaw-start,
smb:::op-WriteX-start,
smb:::op-Write-start
{
        printf("%s clnt=%s uid=0x%x tid=0x%x mid=0x%x\n",
               probename,
               args[0]->ci_remote,
               args[1]->soi_uid,
               args[1]->soi_tid,
               args[1]->soi_mid);
}


smb:::op-Close-done,
smb:::op-CloseAndTreeDisconnect-done,
smb:::op-Transaction-done,
smb:::op-TransactionSecondary-done,
smb:::op-Ioctl-done,
smb:::op-Transaction2-done,
smb:::op-Transaction2Secondary-done,
smb:::op-NtTransact-done,
smb:::op-NtTransactSecondary-done,
smb:::op-Create-done,
smb:::op-CreateNew-done,
smb:::op-CreateTemporary-done,
smb:::op-Delete-done,
smb:::op-CreateDirectory-done,
smb:::op-DeleteDirectory-done,
smb:::op-CheckDirectory-done,
smb:::op-Invalid-done,
smb:::op-Echo-done,
smb:::op-Search-done,
smb:::op-Find-done,
```

```
120 smb:::op-FindClose-done,
121 smb:::op-FindUnique-done,
122 smb:::op-Flush-done,
123 smb:::op-QueryInformationDisk-done,
124 smb:::op-LockByteRange-done,
125 smb:::op-LockingX-done,
126 smb:::op-LogoffX-done,
127 smb:::op-Negotiate-done,
128 smb:::op-NtCancel-done,
129 smb:::op-NtCreateX-done,
130 smb:::op-NtTransactCreate-done,
131 smb:::op-Open-done,
132 smb:::op-OpenX-done,
133 smb:::op-OpenPrintFile-done,
134 smb:::op-ClosePrintFile-done,
135 smb:::op-GetPrintQueue-done,
136 smb:::op-WritePrintFile-done,
137 smb:::op-ProcessExit-done,
138 smb:::op-QueryInformation-done,
139 smb:::op-QueryInformation2-done,
140 smb:::op-Read-done,
141 smb:::op-LockAndRead-done,
142 smb:::op-ReadRaw-done,
143 smb:::op-ReadX-done,
144 smb:::op-Rename-done,
145 smb:::op-NtRename-done,
146 smb:::op-Seek-done,
147 smb:::op-SessionSetupX-done,
148 smb:::op-SetInformation-done,
149 smb:::op-SetInformation2-done,
150 smb:::op-FindClose2-done,
151 smb:::op-TreeConnect-done,
152 smb:::op-TreeConnectX-done,
153 smb:::op-TreeDisconnect-done,
154 smb:::op-UnlockByteRange-done,
155 smb:::op-Write-done,
156 smb:::op-WriteAndClose-done,
157 smb:::op-WriteAndUnlock-done,
158 smb:::op-WriteRaw-done,
159 smb:::op-WriteX-done
160 {
161         printf("%s clnt=%s mid=0x%x status=0x%x\n",
162                probename,
163                args[0]->ci_remote,
164                args[1]->soi_mid,
165                args[1]->soi_status);
166 }
```

```
**********************************************************
    6893 Fri Mar 31 14:29:37 2017
new/usr/src/lib/libdtrace/Makefile.com
NEX-1643 dtrace provider for smbsrv
Also illumos 1841:
 DTrace smb provider was mis-implemented, doesn't exist.
Add back handlers for read/write raw, so that
 legacy dtrace consumers can find the probes.
Kill extra arg in smb_negotiate
Fix missing "done" probe with smb_notify
Add example consumer: smb-trace.d
fix soi_pid
**********************************************************
#
# 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.
# Copyright (c) 2012 by Delphix. All rights reserved.
#

LIBRARY = libdtrace.a
VERS = .1

LIBSRCS = \
        dt_aggregate.c \
        dt_as.c \
        dt_buf.c \
        dt_cc.c \
        dt_cg.c \
        dt_consume.c \
        dt_decl.c \
        dt_dis.c \
        dt_dof.c \
        dt_error.c \
        dt_errtags.c \
        dt_handle.c \
        dt_ident.c \
        dt_inttab.c \
        dt_link.c \
        dt_list.c \
        dt_open.c \
        dt_options.c \
        dt_program.c \
        dt_map.c \
        dt_module.c \
        dt_names.c \
        dt_parser.c \
        dt_pcb.c \
```



```
        dt_pid.c \
        dt_pq.c \
        dt_pragma.c \
        dt_print.c \
        dt_printf.c \
        dt_proc.c \
        dt_provider.c \
        dt_regset.c \
        dt_string.c \
        dt_strtab.c \
        dt_subr.c \
        dt_work.c \
        dt_xlator.c

LIBISASRCS = \
        dt_isadep.c

OBJECTS = dt_lex.o dt_grammar.o $(MACHOBJS) $(LIBSRCS:%.c=%.o) $(LIBISASRCS:%.c=

DRTISRCS = dlink_init.c dlink_common.c
DRTIOBJS = $(DRTISRCS:%.c=pics/%.o)
DRTIOBJ = drti.o

LIBDAUDITSRCS = dlink_audit.c dlink_common.c
LIBDAUDITOBJS = $(LIBDAUDITSRCS:%.c=pics/%.o)
LIBDAUDIT = libdtrace_forceload.so

DLINKSRCS = dlink_common.c dlink_init.c dlink_audit.c

DLIBSRCS += \
        errno.d \
        fc.d \
        io.d \
        ip.d \
        iscsit.d \
        net.d \
        nfs.d \
        nfssrv.d \
        procfs.d \
        regs.d \
        sched.d \
        signal.d \
        scsi.d \
        smb.d \
        srp.d \
        sysevent.d \
        tcp.d \
        udp.d \
        unistd.d

include ../../Makefile.lib

SRCS = $(LIBSRCS:%.c=../common/%.c) $(LIBISASRCS:%.c=../$(MACH)/%.c)
LIBS = $(DYNLIB) $(LINTLIB)

SRCDIR = ../common

CLEANFILES += dt_lex.c dt_grammar.c dt_grammar.h y.output
CLEANFILES += ../common/procfs.sed ../common/procfs.d
CLEANFILES += ../common/io.sed ../common/io.d
CLEANFILES += ../common/ip.sed ../common/ip.d
CLEANFILES += ../common/net.sed ../common/net.d
CLEANFILES += ../common/errno.d ../common/signal.d
CLEANFILES += ../common/dt_errtags.c ../common/dt_names.c
CLEANFILES += ../common/sysevent.sed ../common/sysevent.d
CLEANFILES += ../common/tcp.sed ../common/tcp.d
```

```
120 CLEANFILES += ../common/udp.sed ../common/udp.d
121 CLEANFILES += $(LIBDAUDITOBJS) $(DRTIOBJS)

123 CLOBBERFILES += $(LIBDAUDIT) drti.o

125 CPPFLAGS += -I../common -I.
126 CFLAGS += $(CCVERBOSE) $(C_BIGPICFLAGS)
127 CFLAGS64 += $(CCVERBOSE) $(C_BIGPICFLAGS)

129 CERRWARN += -_gcc=-Wno-unused-label
130 CERRWARN += -_gcc=-Wno-unused-variable
131 CERRWARN += -_gcc=-Wno-parentheses
132 CERRWARN += -_gcc=-Wno-uninitialized
133 CERRWARN += -_gcc=-Wno-switch

135 YYCFLAGS =
136 LDLIBS += -lgen -lproc -lrtld_db -lnsl -lsocket -lctf -lelf -lc
137 DRTILDLIBS = $(LDLIBS.lib) -lc
138 LIBDAUDITLIBS = $(LDLIBS.lib) -lmapmalloc -lc -lproc

140 yydebug := YYCFLAGS += -DYYDEBUG

142 $(LINTLIB) := SRCS = $(SRCDIR)/$(LINTSRC)

144 LFLAGS = -t -v
145 YFLAGS = -d -v

147 ROOTDLIBDIR = $(ROOT)/usr/lib/dtrace
148 ROOTDLIBDIR64 = $(ROOT)/usr/lib/dtrace/64

150 ROOTDLIBS = $(DLIBSRCS:%=$(ROOTDLIBDIR)/%)
151 ROOTDOBJS = $(ROOTDLIBDIR)/$(DRTIOBJ) $(ROOTDLIBDIR)/$(LIBDAUDIT)
152 ROOTDOBJS64 = $(ROOTDLIBDIR64)/$(DRTIOBJ) $(ROOTDLIBDIR64)/$(LIBDAUDIT)

154 $(ROOTDLIBDIR)/%.d := FILEMODE=444
155 $(ROOTDLIBDIR)/%.o := FILEMODE=444
156 $(ROOTDLIBDIR64)/%.o := FILEMODE=444
157 $(ROOTDLIBDIR)/%.so := FILEMODE=555
158 $(ROOTDLIBDIR64)/%.so := FILEMODE=555

160 .KEEP_STATE:

162 all: $(LIBS) $(DRTIOBJ) $(LIBDAUDIT)

164 lint: lintdlink lintcheck

166 lintdlink: $(DLINKSRCS:%.c=../common/%.c)
167         $(LINT.c) $(DLINKSRCS:%.c=../common/%.c) $(DRTILDLIBS)

169 dt_lex.c: $(SRCDIR)/dt_lex.l dt_grammar.h
170         $(LEX) $(LFLAGS) $(SRCDIR)/dt_lex.l > $@

172 dt_grammar.c dt_grammar.h: $(SRCDIR)/dt_grammar.y
173         $(YACC) $(YFLAGS) $(SRCDIR)/dt_grammar.y
174         @mv y.tab.h dt_grammar.h
175         @mv y.tab.c dt_grammar.c

177 pics/dt_lex.o pics/dt_grammar.o := CFLAGS += $(YYCFLAGS)
178 pics/dt_lex.o pics/dt_grammar.o := CFLAGS64 += $(YYCFLAGS)

180 pics/dt_lex.o pics/dt_grammar.o := CERRWARN += -erroff=E_STATEMENT_NOT_REACHED
181 pics/dt_lex.o pics/dt_grammar.o := CCVERBOSE =

183 ../common/dt_errtags.c: ../common/mkerrtags.sh ../common/dt_errtags.h
184         sh ../common/mkerrtags.sh < ../common/dt_errtags.h > $@
```



```
186 ../common/dt_names.c: ../common/mknames.sh $(SRC)/uts/common/sys/dtrace.h
187         sh ../common/mknames.sh < $(SRC)/uts/common/sys/dtrace.h > $@

189 ../common/errno.d: ../common/mkerrno.sh $(SRC)/uts/common/sys/errno.h
190         sh ../common/mkerrno.sh < $(SRC)/uts/common/sys/errno.h > $@

192 ../common/signal.d: ../common/mksignal.sh $(SRC)/uts/common/sys/iso/signal_iso.h
193         sh ../common/mksignal.sh < $(SRC)/uts/common/sys/iso/signal_iso.h > $@

195 ../common/%.sed: ../common/%.sed.in
196         $(COMPILE.cpp) -D_KERNEL $< | tr -d ' ' | tr '"' '@' | \
197             sed 's/\&/\\\&/g' | grep '^s/' > $@

199 ../common/procfs.d: ../common/procfs.sed ../common/procfs.d.in
200         sed -f ../common/procfs.sed < ../common/procfs.d.in > $@

202 ../common/io.d: ../common/io.sed ../common/io.d.in
203         sed -f ../common/io.sed < ../common/io.d.in > $@

205 ../common/ip.d: ../common/ip.sed ../common/ip.d.in
206         sed -f ../common/ip.sed < ../common/ip.d.in > $@

208 ../common/net.d: ../common/net.sed ../common/net.d.in
209         sed -f ../common/net.sed < ../common/net.d.in > $@

211 ../common/sysevent.d: ../common/sysevent.sed ../common/sysevent.d.in
212         sed -f ../common/sysevent.sed < ../common/sysevent.d.in > $@

214 ../common/tcp.d: ..//common/tcp.sed ../common/tcp.d.in
215         sed -f ../common/tcp.sed < ../common/tcp.d.in > $@

217 ../common/udp.d: ../common/udp.sed ../common/udp.d.in
218         sed -f ../common/udp.sed < ../common/udp.d.in > $@

220 pics/%.o: ../$(MACH)/%.c
221         $(COMPILE.c) -o $@ $<
222         $(POST_PROCESS_O)

224 pics/%.o: ../$(MACH)/%.s
225         $(COMPILE.s) -o $@ $<
226         $(POST_PROCESS_O)

228 $(DRTIOBJ): $(DRTIOBJS)
229         $(LD) -o $@ -r -Blocal -Breduce $(DRTIOBJS)
230         $(POST_PROCESS_O)

232 $(LIBDAUDIT): $(LIBDAUDITOBJS)
233         $(LINK.c) -o $@ $(GSHARED) -h$(LIBDAUDIT) $(ZTEXT) $(ZDEFS) $(BDIRECT) \
234             $(MAPFILE.PGA:%=-M%) $(MAPFILE.NED:%=-M%) $(LIBDAUDITOBJS) \
235             $(LIBDAUDITLIBS)
236         $(POST_PROCESS_SO)

238 $(ROOTDLIBDIR):
239         $(INS.dir)

241 $(ROOTDLIBDIR64): $(ROOTDLIBDIR)
242         $(INS.dir)

244 $(ROOTDLIBDIR)/%.d: ../common/%.d
245         $(INS.file)

247 $(ROOTDLIBDIR)/%.d: ../$(MACH)/%.d
248         $(INS.file)

250 $(ROOTDLIBDIR)/%.d: %.d
251         $(INS.file)
```

```
253 $(ROOTDLIBDIR)/%.o: %.o
254         $(INS.file)

256 $(ROOTDLIBDIR64)/%.o: %.o
257         $(INS.file)

259 $(ROOTDLIBDIR)/%.so: %.so
260         $(INS.file)

262 $(ROOTDLIBDIR64)/%.so: %.so
263         $(INS.file)

265 $(ROOTDLIBS): $(ROOTDLIBDIR)

267 $(ROOTDOBJS): $(ROOTDLIBDIR)

269 $(ROOTDOBJS64): $(ROOTDLIBDIR64)

271 include ../../Makefile.targ
```

```
**********************************************************
    3700 Fri Mar 31 14:29:38 2017
new/usr/src/lib/libdtrace/common/smb.d
Build provider 3rd arg from smb_request_t
hacking...
NEX-1643 dtrace provider for smbsrv
Also illumos 1841:
 DTrace smb provider was mis-implemented, doesn't exist.
Add back handlers for read/write raw, so that
 legacy dtrace consumers can find the probes.
Kill extra arg in smb_negotiate
Fix missing "done" probe with smb_notify
Add example consumer: smb-trace.d
fix soi_pid
**********************************************************
```

```
/*
 * CDDL HEADER START
 *
 * The contents of this file are subject to the terms of the
 * Common Development and Distribution License (the "License").
 * You may not use this file except in compliance with the License.
 *
 * You can obtain a copy of the license at usr/src/OPENSOLARIS.LICENSE
 * or http://www.opensolaris.org/os/licensing.
 * See the License for the specific language governing permissions
 * and limitations under the License.
 *
 * When distributing Covered Code, include this CDDL HEADER in each
 * file and include the License file at usr/src/OPENSOLARIS.LICENSE.
 * If applicable, add the following below this CDDL HEADER, with the
 * fields enclosed by brackets "[]" replaced with your own identifying
 * information: Portions Copyright [yyyy] [name of copyright owner]
 *
 * CDDL HEADER END
 */

/*
 * Copyright 2009 Sun Microsystems, Inc.  All rights reserved.
 * Use is subject to license terms.
 *
 * Copyright 2013 Nexenta Systems, Inc.  All rights reserved.
 */

#pragma D depends_on library ip.d
#pragma D depends_on library net.d
#pragma D depends_on module genunix
#pragma D depends_on module smbsrv

#pragma D binding "1.5" translator
translator conninfo_t < struct smb_request *P > {
        ci_protocol =
            P->session->ipaddr.a_family == AF_INET6 ? "tcp6" :
            P->session->ipaddr.a_family == AF_INET ? "tcp" :
            "<unknown>";
        ci_local = "<any>"; /* not interesting */
        ci_remote = P->session->ip_addr_str;
};

/*
 * The smbopinfo_t structure describes the internal form of a
 * single SMB request (SMB v1).
 */
typedef struct smbopinfo {
        cred_t   *soi_cred;             /* credentials for operation */
        string   soi_curpath;           /* file handle path (if any) */
        uint64_t soi_sid;               /* session id */
```



```
        uint32_t soi_pid;               /* process id */
        uint32_t soi_status;            /* status */
        uint16_t soi_tid;               /* tree id */
        uint16_t soi_uid;               /* user id */
        uint16_t soi_mid;               /* request id */
        uint16_t soi_fid;               /* file id */
        uint16_t soi_flags2;            /* flags2 */
        uint8_t  soi_flags;             /* flags */
} smbopinfo_t;

#pragma D binding "1.5" translator
translator smbopinfo_t < struct smb_request *P > {
        soi_cred        = (cred_t *)P->user_cr;
        soi_sid         = P->session->s_kid;
        soi_pid         = P->smb_pid;
        soi_status      = P->smb_error.status;
        soi_tid         = P->smb_tid;
        soi_uid         = P->smb_uid;
        soi_mid         = P->smb_mid;
        soi_fid         = P->smb_fid;
        soi_flags2      = P->smb_flg2;
        soi_flags       = P->smb_flg;

        soi_curpath = (P->fid_ofile == NULL ||
            P->fid_ofile->f_node == NULL ||
            P->fid_ofile->f_node->vp == NULL ||
            P->fid_ofile->f_node->vp->v_path == NULL) ? "<NULL>" :
            P->fid_ofile->f_node->vp->v_path;
};

typedef struct smb_rw_args {
        off_t   soa_offset;
        uint_t  soa_count;
} smb_rw_args_t;

#pragma D binding "1.5" translator
translator smb_rw_args_t < smb_request_t *P > {
        soa_offset = P->arg.rw->rw_offset;
        soa_count  = P->arg.rw->rw_count;
};

typedef struct smb_name_args {
        string  soa_name;
} smb_name_args_t;

#pragma D binding "1.5" translator
translator smb_name_args_t < smb_request_t *P > {
        soa_name = (P->arg.dirop.fqi.fq_path.pn_path == NULL) ? "<NULL>" :
            P->arg.dirop.fqi.fq_path.pn_path;
};

typedef struct smb_open_args {
        string          soa_name;
        uint32_t        soa_desired_access;
        uint32_t        soa_share_access;
        uint32_t        soa_create_options;
        uint32_t        soa_create_disposition;
} smb_open_args_t;

#pragma D binding "1.5" translator
translator smb_open_args_t < smb_request_t *P > {
        soa_name = (P->arg.open.fqi.fq_path.pn_path == NULL) ? "<NULL>" :
            P->arg.open.fqi.fq_path.pn_path;
        soa_desired_access = P->arg.open.desired_access;
        soa_share_access   = P->arg.open.share_access;
        soa_create_options = P->arg.open.create_options;
```

```
118         soa_create_disposition = P->arg.open.create_disposition;
119 };
```

```
**********************************************************
   27392 Fri Mar 31 14:29:38 2017
new/usr/src/pkg/manifests/developer-dtrace.mf
NEX-1643 dtrace provider for smbsrv
Also illumos 1841:
 DTrace smb provider was mis-implemented, doesn't exist.
Add back handlers for read/write raw, so that
 legacy dtrace consumers can find the probes.
Kill extra arg in smb_negotiate
Fix missing "done" probe with smb_notify
Add example consumer: smb-trace.d
fix soi_pid
**********************************************************
```

```
   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 #

  26 #
  27 # Copyright (c) 2012 by Delphix. All rights reserved.
  28 # Copyright 2014 PALO, Richard. All rights reserved.
  29 #

  31 set name=pkg.fmri value=pkg:/developer/dtrace@$(PKGVERS)
  32 set name=pkg.description value="Dynamic Tracing (DTrace) Clients"
  33 set name=pkg.summary value="DTrace Clients"
  34 set name=info.classification \
  35     value=org.opensolaris.category.2008:Development/System
  36 set name=variant.arch value=$(ARCH)
  37 dir path=usr group=sys
  38 dir path=usr/demo
  39 dir path=usr/demo/dtrace
  40 dir path=usr/include
  41 dir path=usr/include/sys
  42 dir path=usr/lib
  43 dir path=usr/lib/$(ARCH64)
  44 dir path=usr/lib/devfsadm group=sys
  45 dir path=usr/lib/devfsadm/linkmod group=sys
  46 dir path=usr/lib/dtrace
  47 dir path=usr/lib/dtrace/64
  48 dir path=usr/lib/mdb group=sys
  49 dir path=usr/lib/mdb/kvm group=sys
  50 dir path=usr/lib/mdb/kvm/$(ARCH64) group=sys
  51 dir path=usr/lib/mdb/raw group=sys
  52 dir path=usr/lib/mdb/raw/$(ARCH64) group=sys
  53 dir path=usr/sbin
```



```
  54 dir path=usr/sbin/$(ARCH32)
  55 dir path=usr/sbin/$(ARCH64)
  56 dir path=usr/share
  57 dir path=usr/share/lib
  58 dir path=usr/share/lib/java group=sys
  59 dir path=usr/share/lib/java/javadoc group=other
  60 dir path=usr/share/lib/java/javadoc/dtrace group=other
  61 dir path=usr/share/lib/java/javadoc/dtrace/api group=other
  62 dir path=usr/share/lib/java/javadoc/dtrace/api/org group=other
  63 dir path=usr/share/lib/java/javadoc/dtrace/api/org/opensolaris group=other
  64 dir path=usr/share/lib/java/javadoc/dtrace/api/org/opensolaris/os group=other
  65 dir path=usr/share/lib/java/javadoc/dtrace/api/org/opensolaris/os/dtrace \
  66     group=other
  67 dir \
  68     path=usr/share/lib/java/javadoc/dtrace/api/org/opensolaris/os/dtrace/class-u
  69     group=other
  70 dir path=usr/share/lib/java/javadoc/dtrace/api/resources group=other
  71 dir path=usr/share/lib/java/javadoc/dtrace/examples group=other
  72 dir path=usr/share/lib/java/javadoc/dtrace/html group=other
  73 dir path=usr/share/lib/java/javadoc/dtrace/images group=other
  74 dir path=usr/share/man/man1m
  75 dir path=usr/share/man/man3lib
  76 file path=usr/demo/dtrace/applicat.d
  77 file path=usr/demo/dtrace/badopen.d
  78 file path=usr/demo/dtrace/begin.d
  79 file path=usr/demo/dtrace/callout.d
  80 file path=usr/demo/dtrace/clause.d
  81 file path=usr/demo/dtrace/clear.d
  82 file path=usr/demo/dtrace/countdown.d
  83 file path=usr/demo/dtrace/counter.d
  84 file path=usr/demo/dtrace/dateprof.d
  85 file path=usr/demo/dtrace/delay.d
  86 file path=usr/demo/dtrace/denorm.d
  87 file path=usr/demo/dtrace/end.d
  88 file path=usr/demo/dtrace/error.d
  89 file path=usr/demo/dtrace/errorpath.d
  90 file path=usr/demo/dtrace/find.d
  91 file path=usr/demo/dtrace/firebird.d
  92 file path=usr/demo/dtrace/hello.d
  93 file path=usr/demo/dtrace/howlong.d
  94 file path=usr/demo/dtrace/index.html
  95 file path=usr/demo/dtrace/interp.d
  96 file path=usr/demo/dtrace/interval.d
  97 file path=usr/demo/dtrace/intr.d
  98 file path=usr/demo/dtrace/iocpu.d
  99 file path=usr/demo/dtrace/iosnoop.d
 100 file path=usr/demo/dtrace/iothrough.d
 101 file path=usr/demo/dtrace/iotime.d
 102 file path=usr/demo/dtrace/ipio.d
 103 file path=usr/demo/dtrace/ipproto.d
 104 $(i386_ONLY)file path=usr/demo/dtrace/iprb.d
 105 file path=usr/demo/dtrace/kstat.d
 106 file path=usr/demo/dtrace/ksyms.d
 107 file path=usr/demo/dtrace/libc.d
 108 file path=usr/demo/dtrace/lquantize.d
 109 file path=usr/demo/dtrace/lwptime.d
 110 file path=usr/demo/dtrace/normalize.d
 111 file path=usr/demo/dtrace/nscd.d
 112 file path=usr/demo/dtrace/pri.d
 113 file path=usr/demo/dtrace/printa.d
 114 file path=usr/demo/dtrace/pritime.d
 115 file path=usr/demo/dtrace/prof.d
 116 file path=usr/demo/dtrace/profpri.d
 117 file path=usr/demo/dtrace/progtime.d
 118 file path=usr/demo/dtrace/putnext.d
 119 file path=usr/demo/dtrace/qlen.d
```

```
120 file path=usr/demo/dtrace/qtime.d
121 file path=usr/demo/dtrace/renormalize.d
122 file path=usr/demo/dtrace/restest.d
123 file path=usr/demo/dtrace/ring.d
124 file path=usr/demo/dtrace/rtime.d
125 file path=usr/demo/dtrace/rwinfo.d
126 file path=usr/demo/dtrace/rwtime.d
127 file path=usr/demo/dtrace/sig.d
128 file path=usr/demo/dtrace/soffice.d
129 file path=usr/demo/dtrace/spec.d
130 file path=usr/demo/dtrace/specopen.d
131 file path=usr/demo/dtrace/ssd.d
132 file path=usr/demo/dtrace/syscall.d
133 file path=usr/demo/dtrace/tcp1stbyte.d
134 file path=usr/demo/dtrace/tcpbytes.d
135 file path=usr/demo/dtrace/tcpbytesstat.d
136 file path=usr/demo/dtrace/tcpconnlat.d
137 file path=usr/demo/dtrace/tcpio.d
138 file path=usr/demo/dtrace/tcpioflags.d
139 file path=usr/demo/dtrace/tcprst.d
140 file path=usr/demo/dtrace/tcpsnoop.d
141 file path=usr/demo/dtrace/tcpstate.d
142 file path=usr/demo/dtrace/tcptop.d
143 file path=usr/demo/dtrace/tick.d
144 file path=usr/demo/dtrace/ticktime.d
145 file path=usr/demo/dtrace/time.d
146 file path=usr/demo/dtrace/tracewrite.d
147 file path=usr/demo/dtrace/trunc.d
148 file path=usr/demo/dtrace/trussrw.d
149 file path=usr/demo/dtrace/udpbytes.d
150 file path=usr/demo/dtrace/udpbytesstat.d
151 file path=usr/demo/dtrace/udpio.d
152 file path=usr/demo/dtrace/udpsnoop.d
153 file path=usr/demo/dtrace/udptop.d
154 file path=usr/demo/dtrace/userfunc.d
155 file path=usr/demo/dtrace/whatfor.d
156 file path=usr/demo/dtrace/whatlock.d
157 file path=usr/demo/dtrace/where.d
158 file path=usr/demo/dtrace/whererun.d
159 file path=usr/demo/dtrace/whoexec.d
160 file path=usr/demo/dtrace/whofor.d
161 file path=usr/demo/dtrace/whoio.d
162 file path=usr/demo/dtrace/whopreempt.d
163 file path=usr/demo/dtrace/whoqueue.d
164 file path=usr/demo/dtrace/whosteal.d
165 file path=usr/demo/dtrace/whowrite.d
166 file path=usr/demo/dtrace/writes.d
167 file path=usr/demo/dtrace/writesbycmd.d
168 file path=usr/demo/dtrace/writesbycmdfd.d
169 file path=usr/demo/dtrace/writetime.d
170 file path=usr/demo/dtrace/writetimeq.d
171 file path=usr/demo/dtrace/xioctl.d
172 file path=usr/demo/dtrace/xterm.d
173 file path=usr/demo/dtrace/xwork.d
174 file path=usr/include/dtrace.h
175 file path=usr/include/sys/dtrace.h
176 file path=usr/include/sys/dtrace_impl.h
177 file path=usr/include/sys/fasttrap.h
178 file path=usr/include/sys/fasttrap_impl.h
179 file path=usr/include/sys/fasttrap_isa.h
180 file path=usr/include/sys/lockstat.h
181 file path=usr/include/sys/sdt.h
182 file path=usr/lib/$(ARCH64)/libdtrace.so.1
183 file path=usr/lib/$(ARCH64)/libdtrace_jni.so.1
184 file path=usr/lib/$(ARCH64)/llib-ldtrace.ln
185 file path=usr/lib/devfsadm/linkmod/SUNW_dtrace_link.so group=sys
```



```
186 file path=usr/lib/dtrace/64/drti.o mode=0444
187 file path=usr/lib/dtrace/64/libdtrace_forceload.so mode=0555
188 file path=usr/lib/dtrace/drti.o mode=0444
189 file path=usr/lib/dtrace/errno.d mode=0444
190 file path=usr/lib/dtrace/fc.d mode=0444
191 file path=usr/lib/dtrace/io.d mode=0444
192 file path=usr/lib/dtrace/ip.d mode=0444
193 file path=usr/lib/dtrace/iscsit.d mode=0444
194 file path=usr/lib/dtrace/libdtrace_forceload.so mode=0555
195 file path=usr/lib/dtrace/net.d mode=0444
196 file path=usr/lib/dtrace/nfs.d mode=0444
197 file path=usr/lib/dtrace/nfssrv.d mode=0444
198 file path=usr/lib/dtrace/procfs.d mode=0444
199 file path=usr/lib/dtrace/regs.d mode=0444
200 file path=usr/lib/dtrace/sched.d mode=0444
201 file path=usr/lib/dtrace/scsi.d mode=0444
202 file path=usr/lib/dtrace/signal.d mode=0444
203 file path=usr/lib/dtrace/smb.d mode=0444
204 file path=usr/lib/dtrace/srp.d mode=0444
205 file path=usr/lib/dtrace/sysevent.d mode=0444
206 file path=usr/lib/dtrace/tcp.d mode=0444
207 file path=usr/lib/dtrace/udp.d mode=0444
208 file path=usr/lib/dtrace/unistd.d mode=0444
209 file path=usr/lib/libdtrace.so.1
210 file path=usr/lib/libdtrace_jni.so.1
211 file path=usr/lib/llib-ldtrace
212 file path=usr/lib/llib-ldtrace.ln
213 file path=usr/lib/mdb/kvm/$(ARCH64)/dtrace.so group=sys mode=0555
214 $(i386_ONLY)file path=usr/lib/mdb/kvm/dtrace.so group=sys mode=0555
215 file path=usr/lib/mdb/raw/$(ARCH64)/dof.so group=sys mode=0555
216 file path=usr/lib/mdb/raw/dof.so group=sys mode=0555
217 file path=usr/sbin/$(ARCH32)/dtrace mode=0555
218 $(i386_ONLY)file path=usr/sbin/$(ARCH32)/intrstat mode=0555
219 $(i386_ONLY)file path=usr/sbin/$(ARCH32)/lockstat mode=0555
220 $(i386_ONLY)file path=usr/sbin/$(ARCH32)/plockstat mode=0555
221 file path=usr/sbin/$(ARCH64)/dtrace mode=0555
222 file path=usr/sbin/$(ARCH64)/intrstat mode=0555
223 file path=usr/sbin/$(ARCH64)/lockstat mode=0555
224 file path=usr/sbin/$(ARCH64)/plockstat mode=0555
225 file path=usr/share/lib/java/dtrace.jar group=sys
226 file path=usr/share/lib/java/javadoc/dtrace/api/allclasses-frame.html \
227     group=other
228 file path=usr/share/lib/java/javadoc/dtrace/api/allclasses-noframe.html \
229     group=other
230 file path=usr/share/lib/java/javadoc/dtrace/api/constant-values.html \
231     group=other
232 file path=usr/share/lib/java/javadoc/dtrace/api/deprecated-list.html \
233     group=other
234 file path=usr/share/lib/java/javadoc/dtrace/api/help-doc.html group=other
235 file path=usr/share/lib/java/javadoc/dtrace/api/index-all.html group=other
236 file path=usr/share/lib/java/javadoc/dtrace/api/index.html group=other
237 file \
238     path=usr/share/lib/java/javadoc/dtrace/api/org/opensolaris/os/dtrace/Aggrega
239     group=other
240 file \
241     path=usr/share/lib/java/javadoc/dtrace/api/org/opensolaris/os/dtrace/Aggrega
242     group=other
243 file \
244     path=usr/share/lib/java/javadoc/dtrace/api/org/opensolaris/os/dtrace/Aggrega
245     group=other
246 file \
247     path=usr/share/lib/java/javadoc/dtrace/api/org/opensolaris/os/dtrace/Aggrega
248     group=other
249 file \
250     path=usr/share/lib/java/javadoc/dtrace/api/org/opensolaris/os/dtrace/AvgValu
251     group=other
```

```
252 file \
253     path=usr/share/lib/java/javadoc/dtrace/api/org/opensolaris/os/dtrace/Consume
254     group=other
255 file \
256     path=usr/share/lib/java/javadoc/dtrace/api/org/opensolaris/os/dtrace/Consume
257     group=other
258 file \
259     path=usr/share/lib/java/javadoc/dtrace/api/org/opensolaris/os/dtrace/Consume
260     group=other
261 file \
262     path=usr/share/lib/java/javadoc/dtrace/api/org/opensolaris/os/dtrace/Consume
263     group=other
264 file \
265     path=usr/share/lib/java/javadoc/dtrace/api/org/opensolaris/os/dtrace/Consume
266     group=other
267 file \
268     path=usr/share/lib/java/javadoc/dtrace/api/org/opensolaris/os/dtrace/Consume
269     group=other
270 file \
271     path=usr/share/lib/java/javadoc/dtrace/api/org/opensolaris/os/dtrace/CountVa
272     group=other
273 file \
274     path=usr/share/lib/java/javadoc/dtrace/api/org/opensolaris/os/dtrace/DTraceE
275     group=other
276 file \
277     path=usr/share/lib/java/javadoc/dtrace/api/org/opensolaris/os/dtrace/DataEve
278     group=other
279 file \
280     path=usr/share/lib/java/javadoc/dtrace/api/org/opensolaris/os/dtrace/Distrib
281     group=other
282 file \
283     path=usr/share/lib/java/javadoc/dtrace/api/org/opensolaris/os/dtrace/Distrib
284     group=other
285 file \
286     path=usr/share/lib/java/javadoc/dtrace/api/org/opensolaris/os/dtrace/Drop.Ki
287     group=other
288 file \
289     path=usr/share/lib/java/javadoc/dtrace/api/org/opensolaris/os/dtrace/Drop.ht
290     group=other
291 file \
292     path=usr/share/lib/java/javadoc/dtrace/api/org/opensolaris/os/dtrace/DropEve
293     group=other
294 file \
295     path=usr/share/lib/java/javadoc/dtrace/api/org/opensolaris/os/dtrace/Error.h
296     group=other
297 file \
298     path=usr/share/lib/java/javadoc/dtrace/api/org/opensolaris/os/dtrace/ErrorEv
299     group=other
300 file \
301     path=usr/share/lib/java/javadoc/dtrace/api/org/opensolaris/os/dtrace/Excepti
302     group=other
303 file \
304     path=usr/share/lib/java/javadoc/dtrace/api/org/opensolaris/os/dtrace/ExitRec
305     group=other
306 file \
307     path=usr/share/lib/java/javadoc/dtrace/api/org/opensolaris/os/dtrace/Flow.Ki
308     group=other
309 file \
310     path=usr/share/lib/java/javadoc/dtrace/api/org/opensolaris/os/dtrace/Flow.ht
311     group=other
312 file \
313     path=usr/share/lib/java/javadoc/dtrace/api/org/opensolaris/os/dtrace/Interfa
314     group=other
315 file \
316     path=usr/share/lib/java/javadoc/dtrace/api/org/opensolaris/os/dtrace/Interfa
317     group=other
```



```
318 file \
319     path=usr/share/lib/java/javadoc/dtrace/api/org/opensolaris/os/dtrace/Interfa
320     group=other
321 file \
322     path=usr/share/lib/java/javadoc/dtrace/api/org/opensolaris/os/dtrace/KernelS
323     group=other
324 file \
325     path=usr/share/lib/java/javadoc/dtrace/api/org/opensolaris/os/dtrace/KernelS
326     group=other
327 file \
328     path=usr/share/lib/java/javadoc/dtrace/api/org/opensolaris/os/dtrace/LinearD
329     group=other
330 file \
331     path=usr/share/lib/java/javadoc/dtrace/api/org/opensolaris/os/dtrace/LocalCo
332     group=other
333 file \
334     path=usr/share/lib/java/javadoc/dtrace/api/org/opensolaris/os/dtrace/LogDist
335     group=other
336 file \
337     path=usr/share/lib/java/javadoc/dtrace/api/org/opensolaris/os/dtrace/LogLine
338     group=other
339 file \
340     path=usr/share/lib/java/javadoc/dtrace/api/org/opensolaris/os/dtrace/MaxValu
341     group=other
342 file \
343     path=usr/share/lib/java/javadoc/dtrace/api/org/opensolaris/os/dtrace/MinValu
344     group=other
345 file \
346     path=usr/share/lib/java/javadoc/dtrace/api/org/opensolaris/os/dtrace/Option.
347     group=other
348 file \
349     path=usr/share/lib/java/javadoc/dtrace/api/org/opensolaris/os/dtrace/PrintaR
350     group=other
351 file \
352     path=usr/share/lib/java/javadoc/dtrace/api/org/opensolaris/os/dtrace/PrintfR
353     group=other
354 file \
355     path=usr/share/lib/java/javadoc/dtrace/api/org/opensolaris/os/dtrace/Probe.h
356     group=other
357 file \
358     path=usr/share/lib/java/javadoc/dtrace/api/org/opensolaris/os/dtrace/ProbeDa
359     group=other
360 file \
361     path=usr/share/lib/java/javadoc/dtrace/api/org/opensolaris/os/dtrace/ProbeDa
362     group=other
363 file \
364     path=usr/share/lib/java/javadoc/dtrace/api/org/opensolaris/os/dtrace/ProbeDe
365     group=other
366 file \
367     path=usr/share/lib/java/javadoc/dtrace/api/org/opensolaris/os/dtrace/ProbeDe
368     group=other
369 file \
370     path=usr/share/lib/java/javadoc/dtrace/api/org/opensolaris/os/dtrace/ProbeIn
371     group=other
372 file \
373     path=usr/share/lib/java/javadoc/dtrace/api/org/opensolaris/os/dtrace/Process
374     group=other
375 file \
376     path=usr/share/lib/java/javadoc/dtrace/api/org/opensolaris/os/dtrace/Process
377     group=other
378 file \
379     path=usr/share/lib/java/javadoc/dtrace/api/org/opensolaris/os/dtrace/Process
380     group=other
381 file \
382     path=usr/share/lib/java/javadoc/dtrace/api/org/opensolaris/os/dtrace/Program
383     group=other
```

```
file \
    path=usr/share/lib/java/javadoc/dtrace/api/org/opensolaris/os/dtrace/Program
    group=other
file \
    path=usr/share/lib/java/javadoc/dtrace/api/org/opensolaris/os/dtrace/Program
    group=other
file \
    path=usr/share/lib/java/javadoc/dtrace/api/org/opensolaris/os/dtrace/Record.
    group=other
file \
    path=usr/share/lib/java/javadoc/dtrace/api/org/opensolaris/os/dtrace/ScalarR
    group=other
file \
    path=usr/share/lib/java/javadoc/dtrace/api/org/opensolaris/os/dtrace/StackFr
    group=other
file \
    path=usr/share/lib/java/javadoc/dtrace/api/org/opensolaris/os/dtrace/StackVa
    group=other
file \
    path=usr/share/lib/java/javadoc/dtrace/api/org/opensolaris/os/dtrace/StddevV
    group=other
file \
    path=usr/share/lib/java/javadoc/dtrace/api/org/opensolaris/os/dtrace/SumValu
    group=other
file \
    path=usr/share/lib/java/javadoc/dtrace/api/org/opensolaris/os/dtrace/SymbolV
    group=other
file \
    path=usr/share/lib/java/javadoc/dtrace/api/org/opensolaris/os/dtrace/Tuple.h
    group=other
file \
    path=usr/share/lib/java/javadoc/dtrace/api/org/opensolaris/os/dtrace/UserSta
    group=other
file \
    path=usr/share/lib/java/javadoc/dtrace/api/org/opensolaris/os/dtrace/UserSym
    group=other
file \
    path=usr/share/lib/java/javadoc/dtrace/api/org/opensolaris/os/dtrace/UserSym
    group=other
file \
    path=usr/share/lib/java/javadoc/dtrace/api/org/opensolaris/os/dtrace/ValueRe
    group=other
file \
    path=usr/share/lib/java/javadoc/dtrace/api/org/opensolaris/os/dtrace/class-u
    group=other
file \
    path=usr/share/lib/java/javadoc/dtrace/api/org/opensolaris/os/dtrace/class-u
    group=other
file \
    path=usr/share/lib/java/javadoc/dtrace/api/org/opensolaris/os/dtrace/class-u
    group=other
file \
    path=usr/share/lib/java/javadoc/dtrace/api/org/opensolaris/os/dtrace/class-u
    group=other
file \
    path=usr/share/lib/java/javadoc/dtrace/api/org/opensolaris/os/dtrace/class-u
    group=other
file \
    path=usr/share/lib/java/javadoc/dtrace/api/org/opensolaris/os/dtrace/class-u
    group=other
file \
    path=usr/share/lib/java/javadoc/dtrace/api/org/opensolaris/os/dtrace/class-u
    group=other
file \
    path=usr/share/lib/java/javadoc/dtrace/api/org/opensolaris/os/dtrace/class-u
    group=other
```



```
file \
    path=usr/share/lib/java/javadoc/dtrace/api/org/opensolaris/os/dtrace/class-u
    group=other
file \
    path=usr/share/lib/java/javadoc/dtrace/api/org/opensolaris/os/dtrace/class-u
    group=other
file \
    path=usr/share/lib/java/javadoc/dtrace/api/org/opensolaris/os/dtrace/class-u
    group=other
file \
    path=usr/share/lib/java/javadoc/dtrace/api/org/opensolaris/os/dtrace/class-u
    group=other
file \
    path=usr/share/lib/java/javadoc/dtrace/api/org/opensolaris/os/dtrace/class-u
    group=other
file \
    path=usr/share/lib/java/javadoc/dtrace/api/org/opensolaris/os/dtrace/class-u
    group=other
file \
    path=usr/share/lib/java/javadoc/dtrace/api/org/opensolaris/os/dtrace/class-u
    group=other
file \
    path=usr/share/lib/java/javadoc/dtrace/api/org/opensolaris/os/dtrace/class-u
    group=other
file \
    path=usr/share/lib/java/javadoc/dtrace/api/org/opensolaris/os/dtrace/class-u
    group=other
file \
    path=usr/share/lib/java/javadoc/dtrace/api/org/opensolaris/os/dtrace/class-u
    group=other
file \
    path=usr/share/lib/java/javadoc/dtrace/api/org/opensolaris/os/dtrace/class-u
    group=other
file \
    path=usr/share/lib/java/javadoc/dtrace/api/org/opensolaris/os/dtrace/class-u
    group=other
file \
    path=usr/share/lib/java/javadoc/dtrace/api/org/opensolaris/os/dtrace/class-u
    group=other
file \
    path=usr/share/lib/java/javadoc/dtrace/api/org/opensolaris/os/dtrace/class-u
    group=other
file \
    path=usr/share/lib/java/javadoc/dtrace/api/org/opensolaris/os/dtrace/class-u
    group=other
file \
    path=usr/share/lib/java/javadoc/dtrace/api/org/opensolaris/os/dtrace/class-u
    group=other
file \
    path=usr/share/lib/java/javadoc/dtrace/api/org/opensolaris/os/dtrace/class-u
    group=other
file \
    path=usr/share/lib/java/javadoc/dtrace/api/org/opensolaris/os/dtrace/class-u
    group=other
file \
    path=usr/share/lib/java/javadoc/dtrace/api/org/opensolaris/os/dtrace/class-u
    group=other
file \
    path=usr/share/lib/java/javadoc/dtrace/api/org/opensolaris/os/dtrace/class-u
    group=other
file \
    path=usr/share/lib/java/javadoc/dtrace/api/org/opensolaris/os/dtrace/class-u
    group=other
file \
    path=usr/share/lib/java/javadoc/dtrace/api/org/opensolaris/os/dtrace/class-u
    group=other
```

```
516 file \
517     path=usr/share/lib/java/javadoc/dtrace/api/org/opensolaris/os/dtrace/class-u
518     group=other
519 file \
520     path=usr/share/lib/java/javadoc/dtrace/api/org/opensolaris/os/dtrace/class-u
521     group=other
522 file \
523     path=usr/share/lib/java/javadoc/dtrace/api/org/opensolaris/os/dtrace/class-u
524     group=other
525 file \
526     path=usr/share/lib/java/javadoc/dtrace/api/org/opensolaris/os/dtrace/class-u
527     group=other
528 file \
529     path=usr/share/lib/java/javadoc/dtrace/api/org/opensolaris/os/dtrace/class-u
530     group=other
531 file \
532     path=usr/share/lib/java/javadoc/dtrace/api/org/opensolaris/os/dtrace/class-u
533     group=other
534 file \
535     path=usr/share/lib/java/javadoc/dtrace/api/org/opensolaris/os/dtrace/class-u
536     group=other
537 file \
538     path=usr/share/lib/java/javadoc/dtrace/api/org/opensolaris/os/dtrace/class-u
539     group=other
540 file \
541     path=usr/share/lib/java/javadoc/dtrace/api/org/opensolaris/os/dtrace/class-u
542     group=other
543 file \
544     path=usr/share/lib/java/javadoc/dtrace/api/org/opensolaris/os/dtrace/class-u
545     group=other
546 file \
547     path=usr/share/lib/java/javadoc/dtrace/api/org/opensolaris/os/dtrace/class-u
548     group=other
549 file \
550     path=usr/share/lib/java/javadoc/dtrace/api/org/opensolaris/os/dtrace/class-u
551     group=other
552 file \
553     path=usr/share/lib/java/javadoc/dtrace/api/org/opensolaris/os/dtrace/class-u
554     group=other
555 file \
556     path=usr/share/lib/java/javadoc/dtrace/api/org/opensolaris/os/dtrace/class-u
557     group=other
558 file \
559     path=usr/share/lib/java/javadoc/dtrace/api/org/opensolaris/os/dtrace/class-u
560     group=other
561 file \
562     path=usr/share/lib/java/javadoc/dtrace/api/org/opensolaris/os/dtrace/class-u
563     group=other
564 file \
565     path=usr/share/lib/java/javadoc/dtrace/api/org/opensolaris/os/dtrace/class-u
566     group=other
567 file \
568     path=usr/share/lib/java/javadoc/dtrace/api/org/opensolaris/os/dtrace/class-u
569     group=other
570 file \
571     path=usr/share/lib/java/javadoc/dtrace/api/org/opensolaris/os/dtrace/class-u
572     group=other
573 file \
574     path=usr/share/lib/java/javadoc/dtrace/api/org/opensolaris/os/dtrace/class-u
575     group=other
576 file \
577     path=usr/share/lib/java/javadoc/dtrace/api/org/opensolaris/os/dtrace/class-u
578     group=other
579 file \
580     path=usr/share/lib/java/javadoc/dtrace/api/org/opensolaris/os/dtrace/class-u
581     group=other
```



```
582 file \
583     path=usr/share/lib/java/javadoc/dtrace/api/org/opensolaris/os/dtrace/class-u
584     group=other
585 file \
586     path=usr/share/lib/java/javadoc/dtrace/api/org/opensolaris/os/dtrace/class-u
587     group=other
588 file \
589     path=usr/share/lib/java/javadoc/dtrace/api/org/opensolaris/os/dtrace/class-u
590     group=other
591 file \
592     path=usr/share/lib/java/javadoc/dtrace/api/org/opensolaris/os/dtrace/class-u
593     group=other
594 file \
595     path=usr/share/lib/java/javadoc/dtrace/api/org/opensolaris/os/dtrace/class-u
596     group=other
597 file \
598     path=usr/share/lib/java/javadoc/dtrace/api/org/opensolaris/os/dtrace/class-u
599     group=other
600 file \
601     path=usr/share/lib/java/javadoc/dtrace/api/org/opensolaris/os/dtrace/class-u
602     group=other
603 file \
604     path=usr/share/lib/java/javadoc/dtrace/api/org/opensolaris/os/dtrace/class-u
605     group=other
606 file \
607     path=usr/share/lib/java/javadoc/dtrace/api/org/opensolaris/os/dtrace/class-u
608     group=other
609 file \
610     path=usr/share/lib/java/javadoc/dtrace/api/org/opensolaris/os/dtrace/class-u
611     group=other
612 file \
613     path=usr/share/lib/java/javadoc/dtrace/api/org/opensolaris/os/dtrace/class-u
614     group=other
615 file \
616     path=usr/share/lib/java/javadoc/dtrace/api/org/opensolaris/os/dtrace/package
617     group=other
618 file \
619     path=usr/share/lib/java/javadoc/dtrace/api/org/opensolaris/os/dtrace/package
620     group=other
621 file \
622     path=usr/share/lib/java/javadoc/dtrace/api/org/opensolaris/os/dtrace/package
623     group=other
624 file \
625     path=usr/share/lib/java/javadoc/dtrace/api/org/opensolaris/os/dtrace/package
626     group=other
627 file path=usr/share/lib/java/javadoc/dtrace/api/overview-tree.html group=other
628 file path=usr/share/lib/java/javadoc/dtrace/api/package-list group=other
629 file path=usr/share/lib/java/javadoc/dtrace/api/resources/background.gif \
630     group=other
631 file path=usr/share/lib/java/javadoc/dtrace/api/resources/tab.gif group=other
632 file path=usr/share/lib/java/javadoc/dtrace/api/resources/titlebar.gif \
633     group=other
634 file path=usr/share/lib/java/javadoc/dtrace/api/resources/titlebar_end.gif \
635     group=other
636 file path=usr/share/lib/java/javadoc/dtrace/api/serialized-form.html \
637     group=other
638 file path=usr/share/lib/java/javadoc/dtrace/api/stylesheet.css group=other
639 file path=usr/share/lib/java/javadoc/dtrace/examples/TestAPI.java group=other
640 file path=usr/share/lib/java/javadoc/dtrace/examples/TestAPI2.java group=other
641 file path=usr/share/lib/java/javadoc/dtrace/examples/TestTarget.java \
642     group=other
643 file path=usr/share/lib/java/javadoc/dtrace/examples/hello.d group=other
644 file path=usr/share/lib/java/javadoc/dtrace/examples/intrstat.d group=other
645 file path=usr/share/lib/java/javadoc/dtrace/examples/syscall.d group=other
646 file path=usr/share/lib/java/javadoc/dtrace/examples/target.d group=other
647 file path=usr/share/lib/java/javadoc/dtrace/html/JavaDTraceAPI.html \
```

```
 648     group=other
 649 file path=usr/share/lib/java/javadoc/dtrace/html/fast.html group=other
 650 file path=usr/share/lib/java/javadoc/dtrace/images/JavaDTraceAPI.gif \
 651     group=other
 652 file path=usr/share/man/man1m/dtrace.1m
 653 file path=usr/share/man/man1m/intrstat.1m
 654 file path=usr/share/man/man1m/lockstat.1m
 655 file path=usr/share/man/man1m/plockstat.1m
 656 file path=usr/share/man/man3lib/libdtrace.3lib
 657 hardlink path=usr/sbin/dtrace target=../../usr/lib/isaexec
 658 hardlink path=usr/sbin/intrstat target=../../usr/lib/isaexec
 659 hardlink path=usr/sbin/lockstat target=../../usr/lib/isaexec
 660 hardlink path=usr/sbin/plockstat target=../../usr/lib/isaexec
 661 legacy pkg=SUNWdtrc desc="Dynamic Tracing (DTrace) Clients" \
 662     name="DTrace Clients"
 663 license cr_Sun license=cr_Sun
 664 license lic_CDDL license=lic_CDDL
 665 link path=usr/lib/$(ARCH64)/libdtrace.so target=libdtrace.so.1
 666 link path=usr/lib/$(ARCH64)/libdtrace_jni.so target=libdtrace_jni.so.1
 667 link path=usr/lib/libdtrace.so target=libdtrace.so.1
 668 link path=usr/lib/libdtrace_jni.so target=libdtrace_jni.so.1
```

```
**********************************************************
    5062 Fri Mar 31 14:29:38 2017
new/usr/src/pkg/manifests/service-file-system-smb.mf
NEX-1643 dtrace provider for smbsrv
Also illumos 1841:
 DTrace smb provider was mis-implemented, doesn't exist.
Add back handlers for read/write raw, so that
 legacy dtrace consumers can find the probes.
Kill extra arg in smb_negotiate
Fix missing "done" probe with smb_notify
Add example consumer: smb-trace.d
fix soi_pid
**********************************************************
#
# 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.
# Copyright 2014 Nexenta Systems, Inc.  All rights reserved.
#

set name=pkg.fmri value=pkg:/service/file-system/smb@$(PKGVERS)
set name=pkg.description value="SMB Server libraries and commands"
set name=pkg.summary value="SMB Server"
set name=info.classification \
    value="org.opensolaris.category.2008:System/File System"
set name=variant.arch value=$(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=lib/svc/manifest/network/smb group=sys
dir path=lib/svc/method
dir path=usr group=sys
dir path=usr/kernel group=sys
dir path=usr/kernel/drv group=sys
dir path=usr/kernel/drv/$(ARCH64) group=sys
dir path=usr/kernel/kmdb group=sys
dir path=usr/kernel/kmdb/$(ARCH64) group=sys
dir path=usr/lib
dir path=usr/lib/fs group=sys
dir path=usr/lib/fs/smb group=sys
dir path=usr/lib/fs/smb/$(ARCH64) group=sys
dir path=usr/lib/mdb group=sys
dir path=usr/lib/mdb/kvm group=sys
dir path=usr/lib/mdb/kvm/$(ARCH64) group=sys
dir path=usr/lib/reparse
dir path=usr/lib/security
```



```
dir path=usr/lib/smbsrv
dir path=usr/lib/smbsrv/dtrace
dir path=usr/sbin
dir path=usr/share/man
dir path=usr/share/man/man1m
dir path=usr/share/man/man4
dir path=usr/share/man/man5
dir path=var group=sys
dir path=var/smb group=sys
dir path=var/smb/cvol group=sys
dir path=var/smb/cvol/windows group=sys
dir path=var/smb/cvol/windows/system32 group=sys
dir path=var/smb/cvol/windows/system32/vss group=sys
dir path=var/svc group=sys
driver name=smbsrv perms="* 0640 root sys"
file path=lib/svc/manifest/network/smb/server.xml group=sys mode=0444
file path=lib/svc/method/svc-smbd mode=0555
file path=usr/kernel/drv/$(ARCH64)/smbsrv group=sys
$(i386_ONLY)file path=usr/kernel/drv/smbsrv group=sys
file path=usr/kernel/drv/smbsrv.conf group=sys
file path=usr/kernel/kmdb/$(ARCH64)/smbsrv group=sys mode=0555
$(i386_ONLY)file path=usr/kernel/kmdb/smbsrv group=sys mode=0555
file path=usr/lib/fs/smb/$(ARCH64)/libshare_smb.so.1
file path=usr/lib/fs/smb/libshare_smb.so.1
file path=usr/lib/mdb/kvm/$(ARCH64)/smbsrv.so group=sys mode=0555
$(i386_ONLY)file path=usr/lib/mdb/kvm/smbsrv.so group=sys mode=0555
file path=usr/lib/reparse/libreparse_smb.so.1
file path=usr/lib/security/pam_smb_passwd.so.1
file path=usr/lib/smbsrv/dtrace/smb-trace.d mode=0555
file path=usr/lib/smbsrv/dtrace/smbd-all.d mode=0555
file path=usr/lib/smbsrv/dtrace/smbd-authsvc.d mode=0555
file path=usr/lib/smbsrv/dtrace/smbd-doorsvc.d mode=0555
file path=usr/lib/smbsrv/dtrace/smbd-pipesvc.d mode=0555
file path=usr/lib/smbsrv/dtrace/smbnode.d mode=0555
file path=usr/lib/smbsrv/dtrace/smbsrv.d mode=0555
file path=usr/lib/smbsrv/dtrace/smbvfs.d mode=0555
file path=usr/lib/smbsrv/libmlrpc.so.1
file path=usr/lib/smbsrv/libmlsvc.so.1
file path=usr/lib/smbsrv/libsmb.so.1
file path=usr/lib/smbsrv/libsmbns.so.1
file path=usr/lib/smbsrv/smbd mode=0555
file path=usr/sbin/smbadm mode=0555
file path=usr/sbin/smbstat mode=0555
file path=usr/share/man/man1m/smbadm.1m
file path=usr/share/man/man1m/smbd.1m
file path=usr/share/man/man1m/smbstat.1m
file path=usr/share/man/man4/smb.4
file path=usr/share/man/man4/smbautohome.4
file path=usr/share/man/man5/pam_smb_passwd.5
file path=var/smb/cvol/windows/system32/eventlog.dll mode=0755
file path=var/smb/smbpasswd group=sys mode=0400 \
    original_name=SUNWsmbs:var/smb/smbpasswd preserve=true
legacy pkg=SUNWsmbskr desc="SMB Server kernel root components" \
    name="SMB Server (Kernel)"
legacy pkg=SUNWsmbsr desc="SMB Server root components" \
    name="SMB Server (Root)"
legacy pkg=SUNWsmbsu desc="SMB Server libraries and commands" \
    name="SMB Server (Usr)"
license cr_Sun license=cr_Sun
license lic_CDDL license=lic_CDDL
license usr/src/uts/common/fs/smbsrv/THIRDPARTYLICENSE.apple \
    license=usr/src/uts/common/fs/smbsrv/THIRDPARTYLICENSE.apple
link path=usr/lib/reparse/libreparse_smb.so target=libreparse_smb.so.1
link path=usr/lib/security/pam_smb_passwd.so target=pam_smb_passwd.so.1
link path=usr/lib/smbsrv/libmlrpc.so target=libmlrpc.so.1
link path=usr/lib/smbsrv/libmlsvc.so target=libmlsvc.so.1
```

```
 120 link path=usr/lib/smbsrv/libsmb.so target=libsmb.so.1
 121 link path=usr/lib/smbsrv/libsmbns.so target=libsmbns.so.1
```

```
**********************************************************
   76595 Fri Mar 31 14:29:38 2017
new/usr/src/uts/common/dtrace/sdt_subr.c
Build provider 3rd arg from smb_request_t
hacking...
NEX-1643 dtrace provider for smbsrv
Also illumos 1841:
 DTrace smb provider was mis-implemented, doesn't exist.
Add back handlers for read/write raw, so that
 legacy dtrace consumers can find the probes.
Kill extra arg in smb_negotiate
Fix missing "done" probe with smb_notify
Add example consumer: smb-trace.d
fix soi_pid
**********************************************************
```

```
/*
 * 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) 2004, 2010, Oracle and/or its affiliates. All rights reserved.
 * Copyright (c) 2012, Joyent, Inc. All rights reserved.
 * Copyright 2013 Nexenta Systems, Inc.  All rights reserved.
 */

#include <sys/sdt_impl.h>

static dtrace_pattr_t vtrace_attr = {
{ DTRACE_STABILITY_UNSTABLE, DTRACE_STABILITY_UNSTABLE, DTRACE_CLASS_ISA },
{ DTRACE_STABILITY_PRIVATE, DTRACE_STABILITY_PRIVATE, DTRACE_CLASS_UNKNOWN },
{ DTRACE_STABILITY_PRIVATE, DTRACE_STABILITY_PRIVATE, DTRACE_CLASS_UNKNOWN },
{ DTRACE_STABILITY_PRIVATE, DTRACE_STABILITY_PRIVATE, DTRACE_CLASS_UNKNOWN },
{ DTRACE_STABILITY_UNSTABLE, DTRACE_STABILITY_UNSTABLE, DTRACE_CLASS_ISA },
};
```

_____*unchanged_portion_omitted_*

```
sdt_provider_t sdt_providers[] = {
        { "vtrace", "__vtrace_", &vtrace_attr },
        { "sysinfo", "__cpu_sysinfo_", &info_attr, DTRACE_PRIV_USER },
        { "vminfo", "__cpu_vminfo_", &info_attr, DTRACE_PRIV_USER },
        { "fpuinfo", "__fpuinfo_", &fpu_attr },
        { "sched", "__sched_", &stab_attr, DTRACE_PRIV_USER },
        { "proc", "__proc_", &stab_attr, DTRACE_PRIV_USER },
        { "io", "__io_", &stab_attr },
        { "ip", "__ip_", &stab_attr },
        { "tcp", "__tcp_", &stab_attr },
        { "udp", "__udp_", &stab_attr },
        { "mib", "__mib_", &stab_attr },
        { "fsinfo", "__fsinfo_", &fsinfo_attr },
        { "iscsi", "__iscsi_", &iscsi_attr },
```



```
        { "nfsv3", "__nfsv3_", &stab_attr },
        { "nfsv4", "__nfsv4_", &stab_attr },
        { "smb", "__smb_", &stab_attr },
        { "xpv", "__xpv_", &xpv_attr },
        { "fc", "__fc_", &fc_attr },
        { "srp", "__srp_", &fc_attr },
        { "sysevent", "__sysevent_", &stab_attr },
        { "sdt", NULL, &sdt_attr },
        { NULL }
};

sdt_argdesc_t sdt_args[] = {
        { "sched", "wakeup", 0, 0, "kthread_t *", "lwpsinfo_t *" },
        { "sched", "wakeup", 1, 0, "kthread_t *", "psinfo_t *" },
        { "sched", "dequeue", 0, 0, "kthread_t *", "lwpsinfo_t *" },
        { "sched", "dequeue", 1, 0, "kthread_t *", "psinfo_t *" },
        { "sched", "dequeue", 2, 1, "disp_t *", "cpuinfo_t *" },
        { "sched", "enqueue", 0, 0, "kthread_t *", "lwpsinfo_t *" },
        { "sched", "enqueue", 1, 0, "kthread_t *", "psinfo_t *" },
        { "sched", "enqueue", 2, 1, "disp_t *", "cpuinfo_t *" },
        { "sched", "enqueue", 3, 2, "int" },
        { "sched", "off-cpu", 0, 0, "kthread_t *", "lwpsinfo_t *" },
        { "sched", "off-cpu", 1, 0, "kthread_t *", "psinfo_t *" },
        { "sched", "tick", 0, 0, "kthread_t *", "lwpsinfo_t *" },
        { "sched", "tick", 1, 0, "kthread_t *", "psinfo_t *" },
        { "sched", "change-pri", 0, 0, "kthread_t *", "lwpsinfo_t *" },
        { "sched", "change-pri", 1, 0, "kthread_t *", "psinfo_t *" },
        { "sched", "change-pri", 2, 1, "pri_t" },
        { "sched", "schedctl-nopreempt", 0, 0, "kthread_t *", "lwpsinfo_t *" },
        { "sched", "schedctl-nopreempt", 1, 0, "kthread_t *", "psinfo_t *" },
        { "sched", "schedctl-nopreempt", 2, 1, "int" },
        { "sched", "schedctl-preempt", 0, 0, "kthread_t *", "lwpsinfo_t *" },
        { "sched", "schedctl-preempt", 1, 0, "kthread_t *", "psinfo_t *" },
        { "sched", "schedctl-yield", 0, 0, "int" },
        { "sched", "surrender", 0, 0, "kthread_t *", "lwpsinfo_t *" },
        { "sched", "surrender", 1, 0, "kthread_t *", "psinfo_t *" },
        { "sched", "cpucaps-sleep", 0, 0, "kthread_t *", "lwpsinfo_t *" },
        { "sched", "cpucaps-sleep", 1, 0, "kthread_t *", "psinfo_t *" },
        { "sched", "cpucaps-wakeup", 0, 0, "kthread_t *", "lwpsinfo_t *" },
        { "sched", "cpucaps-wakeup", 1, 0, "kthread_t *", "psinfo_t *" },

        { "proc", "create", 0, 0, "proc_t *", "psinfo_t *" },
        { "proc", "exec", 0, 0, "string" },
        { "proc", "exec-failure", 0, 0, "int" },
        { "proc", "exit", 0, 0, "int" },
        { "proc", "fault", 0, 0, "int" },
        { "proc", "fault", 1, 1, "siginfo_t *" },
        { "proc", "lwp-create", 0, 0, "kthread_t *", "lwpsinfo_t *" },
        { "proc", "lwp-create", 1, 0, "kthread_t *", "psinfo_t *" },
        { "proc", "signal-clear", 0, 0, "int" },
        { "proc", "signal-clear", 1, 1, "siginfo_t *" },
        { "proc", "signal-discard", 0, 0, "kthread_t *", "lwpsinfo_t *" },
        { "proc", "signal-discard", 1, 1, "proc_t *", "psinfo_t *" },
        { "proc", "signal-discard", 2, 2, "int" },
        { "proc", "signal-handle", 0, 0, "int" },
        { "proc", "signal-handle", 1, 1, "siginfo_t *" },
        { "proc", "signal-handle", 2, 2, "void (*)(void)" },
        { "proc", "signal-send", 0, 0, "kthread_t *", "lwpsinfo_t *" },
        { "proc", "signal-send", 1, 0, "kthread_t *", "psinfo_t *" },
        { "proc", "signal-send", 2, 1, "int" },

        { "io", "start", 0, 0, "buf_t *", "bufinfo_t *" },
        { "io", "start", 1, 0, "buf_t *", "devinfo_t *" },
        { "io", "start", 2, 0, "buf_t *", "fileinfo_t *" },
        { "io", "done", 0, 0, "buf_t *", "bufinfo_t *" },
        { "io", "done", 1, 0, "buf_t *", "devinfo_t *" },
```

```
181         { "io", "done", 2, 0, "buf_t *", "fileinfo_t *" },
182         { "io", "wait-start", 0, 0, "buf_t *", "bufinfo_t *" },
183         { "io", "wait-start", 1, 0, "buf_t *", "devinfo_t *" },
184         { "io", "wait-start", 2, 0, "buf_t *", "fileinfo_t *" },
185         { "io", "wait-done", 0, 0, "buf_t *", "bufinfo_t *" },
186         { "io", "wait-done", 1, 0, "buf_t *", "devinfo_t *" },
187         { "io", "wait-done", 2, 0, "buf_t *", "fileinfo_t *" },

189         { "mib", NULL, 0, 0, "int" },

191         { "fsinfo", NULL, 0, 0, "vnode_t *", "fileinfo_t *" },
192         { "fsinfo", NULL, 1, 1, "int", "int" },

194         { "iscsi", "async-send", 0, 0, "idm_conn_t *", "conninfo_t *" },
195         { "iscsi", "async-send", 1, 1, "iscsi_async_evt_hdr_t *",
196           "iscsiinfo_t *" },
197         { "iscsi", "login-command", 0, 0, "idm_conn_t *", "conninfo_t *" },
198         { "iscsi", "login-command", 1, 1, "iscsi_login_hdr_t *",
199           "iscsiinfo_t *" },
200         { "iscsi", "login-response", 0, 0, "idm_conn_t *", "conninfo_t *" },
201         { "iscsi", "login-response", 1, 1, "iscsi_login_rsp_hdr_t *",
202           "iscsiinfo_t *" },
203         { "iscsi", "logout-command", 0, 0, "idm_conn_t *", "conninfo_t *" },
204         { "iscsi", "logout-command", 1, 1, "iscsi_logout_hdr_t *",
205           "iscsiinfo_t *" },
206         { "iscsi", "logout-response", 0, 0, "idm_conn_t *", "conninfo_t *" },
207         { "iscsi", "logout-response", 1, 1, "iscsi_logout_rsp_hdr_t *",
208           "iscsiinfo_t *" },
209         { "iscsi", "data-request", 0, 0, "idm_conn_t *", "conninfo_t *" },
210         { "iscsi", "data-request", 1, 1, "iscsi_rtt_hdr_t *",
211           "iscsiinfo_t *" },
212         { "iscsi", "data-send", 0, 0, "idm_conn_t *", "conninfo_t *" },
213         { "iscsi", "data-send", 1, 1, "iscsi_data_rsp_hdr_t *",
214           "iscsiinfo_t *" },
215         { "iscsi", "data-receive", 0, 0, "idm_conn_t *", "conninfo_t *" },
216         { "iscsi", "data-receive", 1, 1, "iscsi_data_hdr_t *",
217           "iscsiinfo_t *" },
218         { "iscsi", "nop-send", 0, 0, "idm_conn_t *", "conninfo_t *" },
219         { "iscsi", "nop-send", 1, 1, "iscsi_nop_in_hdr_t *", "iscsiinfo_t *" },
220         { "iscsi", "nop-receive", 0, 0, "idm_conn_t *", "conninfo_t *" },
221         { "iscsi", "nop-receive", 1, 1, "iscsi_nop_out_hdr_t *",
222           "iscsiinfo_t *" },
223         { "iscsi", "scsi-command", 0, 0, "idm_conn_t *", "conninfo_t *" },
224         { "iscsi", "scsi-command", 1, 1, "iscsi_scsi_cmd_hdr_t *",
225           "iscsiinfo_t *" },
226         { "iscsi", "scsi-command", 2, 2, "scsi_task_t *", "scsicmd_t *" },
227         { "iscsi", "scsi-response", 0, 0, "idm_conn_t *", "conninfo_t *" },
228         { "iscsi", "scsi-response", 1, 1, "iscsi_scsi_rsp_hdr_t *",
229           "iscsiinfo_t *" },
230         { "iscsi", "task-command", 0, 0, "idm_conn_t *", "conninfo_t *" },
231         { "iscsi", "task-command", 1, 1, "iscsi_scsi_task_mgt_hdr_t *",
232           "iscsiinfo_t *" },
233         { "iscsi", "task-response", 0, 0, "idm_conn_t *", "conninfo_t *" },
234         { "iscsi", "task-response", 1, 1, "iscsi_scsi_task_mgt_rsp_hdr_t *",
235           "iscsiinfo_t *" },
236         { "iscsi", "text-command", 0, 0, "idm_conn_t *", "conninfo_t *" },
237         { "iscsi", "text-command", 1, 1, "iscsi_text_hdr_t *",
238           "iscsiinfo_t *" },
239         { "iscsi", "text-response", 0, 0, "idm_conn_t *", "conninfo_t *" },
240         { "iscsi", "text-response", 1, 1, "iscsi_text_rsp_hdr_t *",
241           "iscsiinfo_t *" },
242         { "iscsi", "xfer-start", 0, 0, "idm_conn_t *", "conninfo_t *" },
243         { "iscsi", "xfer-start", 1, 0, "idm_conn_t *", "iscsiinfo_t *" },
244         { "iscsi", "xfer-start", 2, 1, "uintptr_t", "xferinfo_t *" },
245         { "iscsi", "xfer-start", 3, 2, "uint32_t"},
246         { "iscsi", "xfer-start", 4, 3, "uintptr_t"},
```



```
247         { "iscsi", "xfer-start", 5, 4, "uint32_t"},
248         { "iscsi", "xfer-start", 6, 5, "uint32_t"},
249         { "iscsi", "xfer-start", 7, 6, "uint32_t"},
250         { "iscsi", "xfer-start", 8, 7, "int"},
251         { "iscsi", "xfer-done", 0, 0, "idm_conn_t *", "conninfo_t *" },
252         { "iscsi", "xfer-done", 1, 0, "idm_conn_t *", "iscsiinfo_t *" },
253         { "iscsi", "xfer-done", 2, 1, "uintptr_t", "xferinfo_t *" },
254         { "iscsi", "xfer-done", 3, 2, "uint32_t"},
255         { "iscsi", "xfer-done", 4, 3, "uintptr_t"},
256         { "iscsi", "xfer-done", 5, 4, "uint32_t"},
257         { "iscsi", "xfer-done", 6, 5, "uint32_t"},
258         { "iscsi", "xfer-done", 7, 6, "uint32_t"},
259         { "iscsi", "xfer-done", 8, 7, "int"},

261         { "nfsv3", "op-getattr-start", 0, 0, "struct svc_req *",
262           "conninfo_t *" },
263         { "nfsv3", "op-getattr-start", 1, 1, "nfsv3oparg_t *",
264           "nfsv3opinfo_t *" },
265         { "nfsv3", "op-getattr-start", 2, 3, "GETATTR3args *" },
266         { "nfsv3", "op-getattr-done", 0, 0, "struct svc_req *",
267           "conninfo_t *" },
268         { "nfsv3", "op-getattr-done", 1, 1, "nfsv3oparg_t *",
269           "nfsv3opinfo_t *" },
270         { "nfsv3", "op-getattr-done", 2, 3, "GETATTR3res *" },
271         { "nfsv3", "op-setattr-start", 0, 0, "struct svc_req *",
272           "conninfo_t *" },
273         { "nfsv3", "op-setattr-start", 1, 1, "nfsv3oparg_t *",
274           "nfsv3opinfo_t *" },
275         { "nfsv3", "op-setattr-start", 2, 3, "SETATTR3args *" },
276         { "nfsv3", "op-setattr-done", 0, 0, "struct svc_req *",
277           "conninfo_t *" },
278         { "nfsv3", "op-setattr-done", 1, 1, "nfsv3oparg_t *",
279           "nfsv3opinfo_t *" },
280         { "nfsv3", "op-setattr-done", 2, 3, "SETATTR3res *" },
281         { "nfsv3", "op-lookup-start", 0, 0, "struct svc_req *",
282           "conninfo_t *" },
283         { "nfsv3", "op-lookup-start", 1, 1, "nfsv3oparg_t *",
284           "nfsv3opinfo_t *" },
285         { "nfsv3", "op-lookup-start", 2, 3, "LOOKUP3args *" },
286         { "nfsv3", "op-lookup-done", 0, 0, "struct svc_req *",
287           "conninfo_t *" },
288         { "nfsv3", "op-lookup-done", 1, 1, "nfsv3oparg_t *",
289           "nfsv3opinfo_t *" },
290         { "nfsv3", "op-lookup-done", 2, 3, "LOOKUP3res *" },
291         { "nfsv3", "op-access-start", 0, 0, "struct svc_req *",
292           "conninfo_t *" },
293         { "nfsv3", "op-access-start", 1, 1, "nfsv3oparg_t *",
294           "nfsv3opinfo_t *" },
295         { "nfsv3", "op-access-start", 2, 3, "ACCESS3args *" },
296         { "nfsv3", "op-access-done", 0, 0, "struct svc_req *",
297           "conninfo_t *" },
298         { "nfsv3", "op-access-done", 1, 1, "nfsv3oparg_t *",
299           "nfsv3opinfo_t *" },
300         { "nfsv3", "op-access-done", 2, 3, "ACCESS3res *" },
301         { "nfsv3", "op-commit-start", 0, 0, "struct svc_req *",
302           "conninfo_t *" },
303         { "nfsv3", "op-commit-start", 1, 1, "nfsv3oparg_t *",
304           "nfsv3opinfo_t *" },
305         { "nfsv3", "op-commit-start", 2, 3, "COMMIT3args *" },
306         { "nfsv3", "op-commit-done", 0, 0, "struct svc_req *",
307           "conninfo_t *" },
308         { "nfsv3", "op-commit-done", 1, 1, "nfsv3oparg_t *",
309           "nfsv3opinfo_t *" },
310         { "nfsv3", "op-commit-done", 2, 3, "COMMIT3res *" },
311         { "nfsv3", "op-create-start", 0, 0, "struct svc_req *",
312           "conninfo_t *" },
```

```
313         { "nfsv3", "op-create-start", 1, 1, "nfsv3oparg_t *",
314             "nfsv3opinfo_t *" },
315         { "nfsv3", "op-create-start", 2, 3, "CREATE3args *" },
316         { "nfsv3", "op-create-done", 0, 0, "struct svc_req *",
317             "conninfo_t *" },
318         { "nfsv3", "op-create-done", 1, 1, "nfsv3oparg_t *",
319             "nfsv3opinfo_t *" },
320         { "nfsv3", "op-create-done", 2, 3, "CREATE3res *" },
321         { "nfsv3", "op-fsinfo-start", 0, 0, "struct svc_req *",
322             "conninfo_t *" },
323         { "nfsv3", "op-fsinfo-start", 1, 1, "nfsv3oparg_t *",
324             "nfsv3opinfo_t *" },
325         { "nfsv3", "op-fsinfo-start", 2, 3, "FSINFO3args *" },
326         { "nfsv3", "op-fsinfo-done", 0, 0, "struct svc_req *",
327             "conninfo_t *" },
328         { "nfsv3", "op-fsinfo-done", 1, 1, "nfsv3oparg_t *",
329             "nfsv3opinfo_t *" },
330         { "nfsv3", "op-fsinfo-done", 2, 3, "FSINFO3res *" },
331         { "nfsv3", "op-fsstat-start", 0, 0, "struct svc_req *",
332             "conninfo_t *" },
333         { "nfsv3", "op-fsstat-start", 1, 1, "nfsv3oparg_t *",
334             "nfsv3opinfo_t *" },
335         { "nfsv3", "op-fsstat-start", 2, 3, "FSSTAT3args *" },
336         { "nfsv3", "op-fsstat-done", 0, 0, "struct svc_req *",
337             "conninfo_t *" },
338         { "nfsv3", "op-fsstat-done", 1, 1, "nfsv3oparg_t *",
339             "nfsv3opinfo_t *" },
340         { "nfsv3", "op-fsstat-done", 2, 3, "FSSTAT3res *" },
341         { "nfsv3", "op-link-start", 0, 0, "struct svc_req *",
342             "conninfo_t *" },
343         { "nfsv3", "op-link-start", 1, 1, "nfsv3oparg_t *",
344             "nfsv3opinfo_t *" },
345         { "nfsv3", "op-link-start", 2, 3, "LINK3args *" },
346         { "nfsv3", "op-link-done", 0, 0, "struct svc_req *",
347             "conninfo_t *" },
348         { "nfsv3", "op-link-done", 1, 1, "nfsv3oparg_t *",
349             "nfsv3opinfo_t *" },
350         { "nfsv3", "op-link-done", 2, 3, "LINK3res *" },
351         { "nfsv3", "op-mkdir-start", 0, 0, "struct svc_req *",
352             "conninfo_t *" },
353         { "nfsv3", "op-mkdir-start", 1, 1, "nfsv3oparg_t *",
354             "nfsv3opinfo_t *" },
355         { "nfsv3", "op-mkdir-start", 2, 3, "MKDIR3args *" },
356         { "nfsv3", "op-mkdir-done", 0, 0, "struct svc_req *",
357             "conninfo_t *" },
358         { "nfsv3", "op-mkdir-done", 1, 1, "nfsv3oparg_t *",
359             "nfsv3opinfo_t *" },
360         { "nfsv3", "op-mkdir-done", 2, 3, "MKDIR3res *" },
361         { "nfsv3", "op-mknod-start", 0, 0, "struct svc_req *",
362             "conninfo_t *" },
363         { "nfsv3", "op-mknod-start", 1, 1, "nfsv3oparg_t *",
364             "nfsv3opinfo_t *" },
365         { "nfsv3", "op-mknod-start", 2, 3, "MKNOD3args *" },
366         { "nfsv3", "op-mknod-done", 0, 0, "struct svc_req *",
367             "conninfo_t *" },
368         { "nfsv3", "op-mknod-done", 1, 1, "nfsv3oparg_t *",
369             "nfsv3opinfo_t *" },
370         { "nfsv3", "op-mknod-done", 2, 3, "MKNOD3res *" },
371         { "nfsv3", "op-null-start", 0, 0, "struct svc_req *",
372             "conninfo_t *" },
373         { "nfsv3", "op-null-start", 1, 1, "nfsv3oparg_t *",
374             "nfsv3opinfo_t *" },
375         { "nfsv3", "op-null-done", 0, 0, "struct svc_req *",
376             "conninfo_t *" },
377         { "nfsv3", "op-null-done", 1, 1, "nfsv3oparg_t *",
378             "nfsv3opinfo_t *" },
```



```
379         { "nfsv3", "op-pathconf-start", 0, 0, "struct svc_req *",
380             "conninfo_t *" },
381         { "nfsv3", "op-pathconf-start", 1, 1, "nfsv3oparg_t *",
382             "nfsv3opinfo_t *" },
383         { "nfsv3", "op-pathconf-start", 2, 3, "PATHCONF3args *" },
384         { "nfsv3", "op-pathconf-done", 0, 0, "struct svc_req *",
385             "conninfo_t *" },
386         { "nfsv3", "op-pathconf-done", 1, 1, "nfsv3oparg_t *",
387             "nfsv3opinfo_t *" },
388         { "nfsv3", "op-pathconf-done", 2, 3, "PATHCONF3res *" },
389         { "nfsv3", "op-read-start", 0, 0, "struct svc_req *",
390             "conninfo_t *" },
391         { "nfsv3", "op-read-start", 1, 1, "nfsv3oparg_t *",
392             "nfsv3opinfo_t *" },
393         { "nfsv3", "op-read-start", 2, 3, "READ3args *" },
394         { "nfsv3", "op-read-done", 0, 0, "struct svc_req *",
395             "conninfo_t *" },
396         { "nfsv3", "op-read-done", 1, 1, "nfsv3oparg_t *",
397             "nfsv3opinfo_t *" },
398         { "nfsv3", "op-read-done", 2, 3, "READ3res *" },
399         { "nfsv3", "op-readdir-start", 0, 0, "struct svc_req *",
400             "conninfo_t *" },
401         { "nfsv3", "op-readdir-start", 1, 1, "nfsv3oparg_t *",
402             "nfsv3opinfo_t *" },
403         { "nfsv3", "op-readdir-start", 2, 3, "READDIR3args *" },
404         { "nfsv3", "op-readdir-done", 0, 0, "struct svc_req *",
405             "conninfo_t *" },
406         { "nfsv3", "op-readdir-done", 1, 1, "nfsv3oparg_t *",
407             "nfsv3opinfo_t *" },
408         { "nfsv3", "op-readdir-done", 2, 3, "READDIR3res *" },
409         { "nfsv3", "op-readdirplus-start", 0, 0, "struct svc_req *",
410             "conninfo_t *" },
411         { "nfsv3", "op-readdirplus-start", 1, 1, "nfsv3oparg_t *",
412             "nfsv3opinfo_t *" },
413         { "nfsv3", "op-readdirplus-start", 2, 3, "READDIRPLUS3args *" },
414         { "nfsv3", "op-readdirplus-done", 0, 0, "struct svc_req *",
415             "conninfo_t *" },
416         { "nfsv3", "op-readdirplus-done", 1, 1, "nfsv3oparg_t *",
417             "nfsv3opinfo_t *" },
418         { "nfsv3", "op-readdirplus-done", 2, 3, "READDIRPLUS3res *" },
419         { "nfsv3", "op-readlink-start", 0, 0, "struct svc_req *",
420             "conninfo_t *" },
421         { "nfsv3", "op-readlink-start", 1, 1, "nfsv3oparg_t *",
422             "nfsv3opinfo_t *" },
423         { "nfsv3", "op-readlink-start", 2, 3, "READLINK3args *" },
424         { "nfsv3", "op-readlink-done", 0, 0, "struct svc_req *",
425             "conninfo_t *" },
426         { "nfsv3", "op-readlink-done", 1, 1, "nfsv3oparg_t *",
427             "nfsv3opinfo_t *" },
428         { "nfsv3", "op-readlink-done", 2, 3, "READLINK3res *" },
429         { "nfsv3", "op-remove-start", 0, 0, "struct svc_req *",
430             "conninfo_t *" },
431         { "nfsv3", "op-remove-start", 1, 1, "nfsv3oparg_t *",
432             "nfsv3opinfo_t *" },
433         { "nfsv3", "op-remove-start", 2, 3, "REMOVE3args *" },
434         { "nfsv3", "op-remove-done", 0, 0, "struct svc_req *",
435             "conninfo_t *" },
436         { "nfsv3", "op-remove-done", 1, 1, "nfsv3oparg_t *",
437             "nfsv3opinfo_t *" },
438         { "nfsv3", "op-remove-done", 2, 3, "REMOVE3res *" },
439         { "nfsv3", "op-rename-start", 0, 0, "struct svc_req *",
440             "conninfo_t *" },
441         { "nfsv3", "op-rename-start", 1, 1, "nfsv3oparg_t *",
442             "nfsv3opinfo_t *" },
443         { "nfsv3", "op-rename-start", 2, 3, "RENAME3args *" },
444         { "nfsv3", "op-rename-done", 0, 0, "struct svc_req *",
```

```
445             "conninfo_t *" },
446         { "nfsv3", "op-rename-done", 1, 1, "nfsv3oparg_t *",
447             "nfsv3opinfo_t *" },
448         { "nfsv3", "op-rename-done", 2, 3, "RENAME3res *" },
449         { "nfsv3", "op-rmdir-start", 0, 0, "struct svc_req *",
450             "conninfo_t *" },
451         { "nfsv3", "op-rmdir-start", 1, 1, "nfsv3oparg_t *",
452             "nfsv3opinfo_t *" },
453         { "nfsv3", "op-rmdir-start", 2, 3, "RMDIR3args *" },
454         { "nfsv3", "op-rmdir-done", 0, 0, "struct svc_req *",
455             "conninfo_t *" },
456         { "nfsv3", "op-rmdir-done", 1, 1, "nfsv3oparg_t *",
457             "nfsv3opinfo_t *" },
458         { "nfsv3", "op-rmdir-done", 2, 3, "RMDIR3res *" },
459         { "nfsv3", "op-setattr-start", 0, 0, "struct svc_req *",
460             "conninfo_t *" },
461         { "nfsv3", "op-setattr-start", 1, 1, "nfsv3oparg_t *",
462             "nfsv3opinfo_t *" },
463         { "nfsv3", "op-setattr-start", 2, 3, "SETATTR3args *" },
464         { "nfsv3", "op-setattr-done", 0, 0, "struct svc_req *",
465             "conninfo_t *" },
466         { "nfsv3", "op-setattr-done", 1, 1, "nfsv3oparg_t *",
467             "nfsv3opinfo_t *" },
468         { "nfsv3", "op-setattr-done", 2, 3, "SETATTR3res *" },
469         { "nfsv3", "op-symlink-start", 0, 0, "struct svc_req *",
470             "conninfo_t *" },
471         { "nfsv3", "op-symlink-start", 1, 1, "nfsv3oparg_t *",
472             "nfsv3opinfo_t *" },
473         { "nfsv3", "op-symlink-start", 2, 3, "SYMLINK3args *" },
474         { "nfsv3", "op-symlink-done", 0, 0, "struct svc_req *",
475             "conninfo_t *" },
476         { "nfsv3", "op-symlink-done", 1, 1, "nfsv3oparg_t *",
477             "nfsv3opinfo_t *" },
478         { "nfsv3", "op-symlink-done", 2, 3, "SYMLINK3res *" },
479         { "nfsv3", "op-write-start", 0, 0, "struct svc_req *",
480             "conninfo_t *" },
481         { "nfsv3", "op-write-start", 1, 1, "nfsv3oparg_t *",
482             "nfsv3opinfo_t *" },
483         { "nfsv3", "op-write-start", 2, 3, "WRITE3args *" },
484         { "nfsv3", "op-write-done", 0, 0, "struct svc_req *",
485             "conninfo_t *" },
486         { "nfsv3", "op-write-done", 1, 1, "nfsv3oparg_t *",
487             "nfsv3opinfo_t *" },
488         { "nfsv3", "op-write-done", 2, 3, "WRITE3res *" },

490         { "nfsv4", "null-start", 0, 0, "struct svc_req *", "conninfo_t *" },
491         { "nfsv4", "null-done", 0, 0, "struct svc_req *", "conninfo_t *" },
492         { "nfsv4", "compound-start", 0, 0, "struct compound_state *",
493             "conninfo_t *" },
494         { "nfsv4", "compound-start", 1, 0, "struct compound_state *",
495             "nfsv4opinfo_t *" },
496         { "nfsv4", "compound-start", 2, 1, "COMPOUND4args *" },
497         { "nfsv4", "compound-done", 0, 0, "struct compound_state *",
498             "conninfo_t *" },
499         { "nfsv4", "compound-done", 1, 0, "struct compound_state *",
500             "nfsv4opinfo_t *" },
501         { "nfsv4", "compound-done", 2, 1, "COMPOUND4res *" },
502         { "nfsv4", "op-access-start", 0, 0, "struct compound_state *",
503             "conninfo_t *"},
504         { "nfsv4", "op-access-start", 1, 0, "struct compound_state *",
505             "nfsv4opinfo_t *" },
506         { "nfsv4", "op-access-start", 2, 1, "ACCESS4args *" },
507         { "nfsv4", "op-access-done", 0, 0, "struct compound_state *",
508             "conninfo_t *" },
509         { "nfsv4", "op-access-done", 1, 0, "struct compound_state *",
510             "nfsv4opinfo_t *" },
```



```
511         { "nfsv4", "op-access-done", 2, 1, "ACCESS4res *" },
512         { "nfsv4", "op-close-start", 0, 0, "struct compound_state *",
513             "conninfo_t *" },
514         { "nfsv4", "op-close-start", 1, 0, "struct compound_state *",
515             "nfsv4opinfo_t *" },
516         { "nfsv4", "op-close-start", 2, 1, "CLOSE4args *" },
517         { "nfsv4", "op-close-done", 0, 0, "struct compound_state *",
518             "conninfo_t *" },
519         { "nfsv4", "op-close-done", 1, 0, "struct compound_state *",
520             "nfsv4opinfo_t *" },
521         { "nfsv4", "op-close-done", 2, 1, "CLOSE4res *" },
522         { "nfsv4", "op-commit-start", 0, 0, "struct compound_state *",
523             "conninfo_t *" },
524         { "nfsv4", "op-commit-start", 1, 0, "struct compound_state *",
525             "nfsv4opinfo_t *" },
526         { "nfsv4", "op-commit-start", 2, 1, "COMMIT4args *" },
527         { "nfsv4", "op-commit-done", 0, 0, "struct compound_state *",
528             "conninfo_t *" },
529         { "nfsv4", "op-commit-done", 1, 0, "struct compound_state *",
530             "nfsv4opinfo_t *" },
531         { "nfsv4", "op-commit-done", 2, 1, "COMMIT4res *" },
532         { "nfsv4", "op-create-start", 0, 0, "struct compound_state *",
533             "conninfo_t *" },
534         { "nfsv4", "op-create-start", 1, 0, "struct compound_state *",
535             "nfsv4opinfo_t *" },
536         { "nfsv4", "op-create-start", 2, 1, "CREATE4args *" },
537         { "nfsv4", "op-create-done", 0, 0, "struct compound_state *",
538             "conninfo_t *" },
539         { "nfsv4", "op-create-done", 1, 0, "struct compound_state *",
540             "nfsv4opinfo_t *" },
541         { "nfsv4", "op-create-done", 2, 1, "CREATE4res *" },
542         { "nfsv4", "op-delegpurge-start", 0, 0, "struct compound_state *",
543             "conninfo_t *" },
544         { "nfsv4", "op-delegpurge-start", 1, 0, "struct compound_state *",
545             "nfsv4opinfo_t *" },
546         { "nfsv4", "op-delegpurge-start", 2, 1, "DELEGPURGE4args *" },
547         { "nfsv4", "op-delegpurge-done", 0, 0, "struct compound_state *",
548             "conninfo_t *" },
549         { "nfsv4", "op-delegpurge-done", 1, 0, "struct compound_state *",
550             "nfsv4opinfo_t *" },
551         { "nfsv4", "op-delegpurge-done", 2, 1, "DELEGPURGE4res *" },
552         { "nfsv4", "op-delegreturn-start", 0, 0, "struct compound_state *",
553             "conninfo_t *" },
554         { "nfsv4", "op-delegreturn-start", 1, 0, "struct compound_state *",
555             "nfsv4opinfo_t *" },
556         { "nfsv4", "op-delegreturn-start", 2, 1, "DELEGRETURN4args *" },
557         { "nfsv4", "op-delegreturn-done", 0, 0, "struct compound_state *",
558             "conninfo_t *" },
559         { "nfsv4", "op-delegreturn-done", 1, 0, "struct compound_state *",
560             "nfsv4opinfo_t *" },
561         { "nfsv4", "op-delegreturn-done", 2, 1, "DELEGRETURN4res *" },
562         { "nfsv4", "op-getattr-start", 0, 0, "struct compound_state *",
563             "conninfo_t *" },
564         { "nfsv4", "op-getattr-start", 1, 0, "struct compound_state *",
565             "nfsv4opinfo_t *" },
566         { "nfsv4", "op-getattr-start", 2, 1, "GETATTR4args *" },
567         { "nfsv4", "op-getattr-done", 0, 0, "struct compound_state *",
568             "conninfo_t *" },
569         { "nfsv4", "op-getattr-done", 1, 0, "struct compound_state *",
570             "nfsv4opinfo_t *" },
571         { "nfsv4", "op-getattr-done", 2, 1, "GETATTR4res *" },
572         { "nfsv4", "op-getfh-start", 0, 0, "struct compound_state *",
573             "conninfo_t *" },
574         { "nfsv4", "op-getfh-start", 1, 0, "struct compound_state *",
575             "nfsv4opinfo_t *" },
576         { "nfsv4", "op-getfh-done", 0, 0, "struct compound_state *",
```

```
577             "conninfo_t *" },
578         { "nfsv4", "op-getfh-done", 1, 0, "struct compound_state *",
579             "nfsv4opinfo_t *" },
580         { "nfsv4", "op-getfh-done", 2, 1, "GETFH4res *" },
581         { "nfsv4", "op-link-start", 0, 0, "struct compound_state *",
582             "conninfo_t *" },
583         { "nfsv4", "op-link-start", 1, 0, "struct compound_state *",
584             "nfsv4opinfo_t *" },
585         { "nfsv4", "op-link-start", 2, 1, "LINK4args *" },
586         { "nfsv4", "op-link-done", 0, 0, "struct compound_state *",
587             "conninfo_t *" },
588         { "nfsv4", "op-link-done", 1, 0, "struct compound_state *",
589             "nfsv4opinfo_t *" },
590         { "nfsv4", "op-link-done", 2, 1, "LINK4res *" },
591         { "nfsv4", "op-lock-start", 0, 0, "struct compound_state *",
592             "conninfo_t *" },
593         { "nfsv4", "op-lock-start", 1, 0, "struct compound_state *",
594             "nfsv4opinfo_t *" },
595         { "nfsv4", "op-lock-start", 2, 1, "LOCK4args *" },
596         { "nfsv4", "op-lock-done", 0, 0, "struct compound_state *",
597             "conninfo_t *" },
598         { "nfsv4", "op-lock-done", 1, 0, "struct compound_state *",
599             "nfsv4opinfo_t *" },
600         { "nfsv4", "op-lock-done", 2, 1, "LOCK4res *" },
601         { "nfsv4", "op-lockt-start", 0, 0, "struct compound_state *",
602             "conninfo_t *" },
603         { "nfsv4", "op-lockt-start", 1, 0, "struct compound_state *",
604             "nfsv4opinfo_t *" },
605         { "nfsv4", "op-lockt-start", 2, 1, "LOCKT4args *" },
606         { "nfsv4", "op-lockt-done", 0, 0, "struct compound_state *",
607             "conninfo_t *" },
608         { "nfsv4", "op-lockt-done", 1, 0, "struct compound_state *",
609             "nfsv4opinfo_t *" },
610         { "nfsv4", "op-lockt-done", 2, 1, "LOCKT4res *" },
611         { "nfsv4", "op-locku-start", 0, 0, "struct compound_state *",
612             "conninfo_t *" },
613         { "nfsv4", "op-locku-start", 1, 0, "struct compound_state *",
614             "nfsv4opinfo_t *" },
615         { "nfsv4", "op-locku-start", 2, 1, "LOCKU4args *" },
616         { "nfsv4", "op-locku-done", 0, 0, "struct compound_state *",
617             "conninfo_t *" },
618         { "nfsv4", "op-locku-done", 1, 0, "struct compound_state *",
619             "nfsv4opinfo_t *" },
620         { "nfsv4", "op-locku-done", 2, 1, "LOCKU4res *" },
621         { "nfsv4", "op-lookup-start", 0, 0, "struct compound_state *",
622             "conninfo_t *" },
623         { "nfsv4", "op-lookup-start", 1, 0, "struct compound_state *",
624             "nfsv4opinfo_t *" },
625         { "nfsv4", "op-lookup-start", 2, 1, "LOOKUP4args *" },
626         { "nfsv4", "op-lookup-done", 0, 0, "struct compound_state *",
627             "conninfo_t *" },
628         { "nfsv4", "op-lookup-done", 1, 0, "struct compound_state *",
629             "nfsv4opinfo_t *" },
630         { "nfsv4", "op-lookup-done", 2, 1, "LOOKUP4res *" },
631         { "nfsv4", "op-lookupp-start", 0, 0, "struct compound_state *",
632             "conninfo_t *" },
633         { "nfsv4", "op-lookupp-start", 1, 0, "struct compound_state *",
634             "nfsv4opinfo_t *" },
635         { "nfsv4", "op-lookupp-done", 0, 0, "struct compound_state *",
636             "conninfo_t *" },
637         { "nfsv4", "op-lookupp-done", 1, 0, "struct compound_state *",
638             "nfsv4opinfo_t *" },
639         { "nfsv4", "op-lookupp-done", 2, 1, "LOOKUPP4res *" },
640         { "nfsv4", "op-nverify-start", 0, 0, "struct compound_state *",
641             "conninfo_t *" },
642         { "nfsv4", "op-nverify-start", 1, 0, "struct compound_state *",
```



```
643             "nfsv4opinfo_t *" },
644         { "nfsv4", "op-nverify-start", 2, 1, "NVERIFY4args *" },
645         { "nfsv4", "op-nverify-done", 0, 0, "struct compound_state *",
646             "conninfo_t *" },
647         { "nfsv4", "op-nverify-done", 1, 0, "struct compound_state *",
648             "nfsv4opinfo_t *" },
649         { "nfsv4", "op-nverify-done", 2, 1, "NVERIFY4res *" },
650         { "nfsv4", "op-open-start", 0, 0, "struct compound_state *",
651             "conninfo_t *" },
652         { "nfsv4", "op-open-start", 1, 0, "struct compound_state *",
653             "nfsv4opinfo_t *" },
654         { "nfsv4", "op-open-start", 2, 1, "OPEN4args *" },
655         { "nfsv4", "op-open-done", 0, 0, "struct compound_state *",
656             "conninfo_t *" },
657         { "nfsv4", "op-open-done", 1, 0, "struct compound_state *",
658             "nfsv4opinfo_t *" },
659         { "nfsv4", "op-open-done", 2, 1, "OPEN4res *" },
660         { "nfsv4", "op-open-confirm-start", 0, 0, "struct compound_state *",
661             "conninfo_t *" },
662         { "nfsv4", "op-open-confirm-start", 1, 0, "struct compound_state *",
663             "nfsv4opinfo_t *" },
664         { "nfsv4", "op-open-confirm-start", 2, 1, "OPEN_CONFIRM4args *" },
665         { "nfsv4", "op-open-confirm-done", 0, 0, "struct compound_state *",
666             "conninfo_t *" },
667         { "nfsv4", "op-open-confirm-done", 1, 0, "struct compound_state *",
668             "nfsv4opinfo_t *" },
669         { "nfsv4", "op-open-confirm-done", 2, 1, "OPEN_CONFIRM4res *" },
670         { "nfsv4", "op-open-downgrade-start", 0, 0, "struct compound_state *",
671             "conninfo_t *" },
672         { "nfsv4", "op-open-downgrade-start", 1, 0, "struct compound_state *",
673             "nfsv4opinfo_t *" },
674         { "nfsv4", "op-open-downgrade-start", 2, 1, "OPEN_DOWNGRADE4args *" },
675         { "nfsv4", "op-open-downgrade-done", 0, 0, "struct compound_state *",
676             "conninfo_t *" },
677         { "nfsv4", "op-open-downgrade-done", 1, 0, "struct compound_state *",
678             "nfsv4opinfo_t *" },
679         { "nfsv4", "op-open-downgrade-done", 2, 1, "OPEN_DOWNGRADE4res *" },
680         { "nfsv4", "op-openattr-start", 0, 0, "struct compound_state *",
681             "conninfo_t *" },
682         { "nfsv4", "op-openattr-start", 1, 0, "struct compound_state *",
683             "nfsv4opinfo_t *" },
684         { "nfsv4", "op-openattr-start", 2, 1, "OPENATTR4args *" },
685         { "nfsv4", "op-openattr-done", 0, 0, "struct compound_state *",
686             "conninfo_t *" },
687         { "nfsv4", "op-openattr-done", 1, 0, "struct compound_state *",
688             "nfsv4opinfo_t *" },
689         { "nfsv4", "op-openattr-done", 2, 1, "OPENATTR4res *" },
690         { "nfsv4", "op-putfh-start", 0, 0, "struct compound_state *",
691             "conninfo_t *" },
692         { "nfsv4", "op-putfh-start", 1, 0, "struct compound_state *",
693             "nfsv4opinfo_t *" },
694         { "nfsv4", "op-putfh-start", 2, 1, "PUTFH4args *" },
695         { "nfsv4", "op-putfh-done", 0, 0, "struct compound_state *",
696             "conninfo_t *" },
697         { "nfsv4", "op-putfh-done", 1, 0, "struct compound_state *",
698             "nfsv4opinfo_t *" },
699         { "nfsv4", "op-putfh-done", 2, 1, "PUTFH4res *" },
700         { "nfsv4", "op-putpubfh-start", 0, 0, "struct compound_state *",
701             "conninfo_t *" },
702         { "nfsv4", "op-putpubfh-start", 1, 0, "struct compound_state *",
703             "nfsv4opinfo_t *" },
704         { "nfsv4", "op-putpubfh-done", 0, 0, "struct compound_state *",
705             "conninfo_t *" },
706         { "nfsv4", "op-putpubfh-done", 1, 0, "struct compound_state *",
707             "nfsv4opinfo_t *" },
708         { "nfsv4", "op-putpubfh-done", 2, 1, "PUTPUBFH4res *" },
```

```
709         { "nfsv4", "op-putrootfh-start", 0, 0, "struct compound_state *",
710           "conninfo_t *" },
711         { "nfsv4", "op-putrootfh-start", 1, 0, "struct compound_state *",
712           "nfsv4opinfo_t *" },
713         { "nfsv4", "op-putrootfh-done", 0, 0, "struct compound_state *",
714           "conninfo_t *" },
715         { "nfsv4", "op-putrootfh-done", 1, 0, "struct compound_state *",
716           "nfsv4opinfo_t *" },
717         { "nfsv4", "op-putrootfh-done", 2, 1, "PUTROOTFH4res *" },
718         { "nfsv4", "op-read-start", 0, 0, "struct compound_state *",
719           "conninfo_t *" },
720         { "nfsv4", "op-read-start", 1, 0, "struct compound_state *",
721           "nfsv4opinfo_t *" },
722         { "nfsv4", "op-read-start", 2, 1, "READ4args *" },
723         { "nfsv4", "op-read-done", 0, 0, "struct compound_state *",
724           "conninfo_t *" },
725         { "nfsv4", "op-read-done", 1, 0, "struct compound_state *",
726           "nfsv4opinfo_t *" },
727         { "nfsv4", "op-read-done", 2, 1, "READ4res *" },
728         { "nfsv4", "op-readdir-start", 0, 0, "struct compound_state *",
729           "conninfo_t *" },
730         { "nfsv4", "op-readdir-start", 1, 0, "struct compound_state *",
731           "nfsv4opinfo_t *" },
732         { "nfsv4", "op-readdir-start", 2, 1, "READDIR4args *" },
733         { "nfsv4", "op-readdir-done", 0, 0, "struct compound_state *",
734           "conninfo_t *" },
735         { "nfsv4", "op-readdir-done", 1, 0, "struct compound_state *",
736           "nfsv4opinfo_t *" },
737         { "nfsv4", "op-readdir-done", 2, 1, "READDIR4res *" },
738         { "nfsv4", "op-readlink-start", 0, 0, "struct compound_state *",
739           "conninfo_t *" },
740         { "nfsv4", "op-readlink-start", 1, 0, "struct compound_state *",
741           "nfsv4opinfo_t *" },
742         { "nfsv4", "op-readlink-done", 0, 0, "struct compound_state *",
743           "conninfo_t *" },
744         { "nfsv4", "op-readlink-done", 1, 0, "struct compound_state *",
745           "nfsv4opinfo_t *" },
746         { "nfsv4", "op-readlink-done", 2, 1, "READLINK4res *" },
747         { "nfsv4", "op-release-lockowner-start", 0, 0,
748           "struct compound_state *", "conninfo_t *" },
749         { "nfsv4", "op-release-lockowner-start", 1, 0,
750           "struct compound_state *", "nfsv4opinfo_t *" },
751         { "nfsv4", "op-release-lockowner-start", 2, 1,
752           "RELEASE_LOCKOWNER4args *" },
753         { "nfsv4", "op-release-lockowner-done", 0, 0,
754           "struct compound_state *", "conninfo_t *" },
755         { "nfsv4", "op-release-lockowner-done", 1, 0,
756           "struct compound_state *", "nfsv4opinfo_t *" },
757         { "nfsv4", "op-release-lockowner-done", 2, 1,
758           "RELEASE_LOCKOWNER4res *" },
759         { "nfsv4", "op-remove-start", 0, 0, "struct compound_state *",
760           "conninfo_t *" },
761         { "nfsv4", "op-remove-start", 1, 0, "struct compound_state *",
762           "nfsv4opinfo_t *" },
763         { "nfsv4", "op-remove-start", 2, 1, "REMOVE4args *" },
764         { "nfsv4", "op-remove-done", 0, 0, "struct compound_state *",
765           "conninfo_t *" },
766         { "nfsv4", "op-remove-done", 1, 0, "struct compound_state *",
767           "nfsv4opinfo_t *" },
768         { "nfsv4", "op-remove-done", 2, 1, "REMOVE4res *" },
769         { "nfsv4", "op-rename-start", 0, 0, "struct compound_state *",
770           "conninfo_t *" },
771         { "nfsv4", "op-rename-start", 1, 0, "struct compound_state *",
772           "nfsv4opinfo_t *" },
773         { "nfsv4", "op-rename-start", 2, 1, "RENAME4args *" },
774         { "nfsv4", "op-rename-done", 0, 0, "struct compound_state *",
```



```
775           "conninfo_t *" },
776         { "nfsv4", "op-rename-done", 1, 0, "struct compound_state *",
777           "nfsv4opinfo_t *" },
778         { "nfsv4", "op-rename-done", 2, 1, "RENAME4res *" },
779         { "nfsv4", "op-renew-start", 0, 0, "struct compound_state *",
780           "conninfo_t *" },
781         { "nfsv4", "op-renew-start", 1, 0, "struct compound_state *",
782           "nfsv4opinfo_t *" },
783         { "nfsv4", "op-renew-start", 2, 1, "RENEW4args *" },
784         { "nfsv4", "op-renew-done", 0, 0, "struct compound_state *",
785           "conninfo_t *" },
786         { "nfsv4", "op-renew-done", 1, 0, "struct compound_state *",
787           "nfsv4opinfo_t *" },
788         { "nfsv4", "op-renew-done", 2, 1, "RENEW4res *" },
789         { "nfsv4", "op-restorefh-start", 0, 0, "struct compound_state *",
790           "conninfo_t *" },
791         { "nfsv4", "op-restorefh-start", 1, 0, "struct compound_state *",
792           "nfsv4opinfo_t *" },
793         { "nfsv4", "op-restorefh-done", 0, 0, "struct compound_state *",
794           "conninfo_t *" },
795         { "nfsv4", "op-restorefh-done", 1, 0, "struct compound_state *",
796           "nfsv4opinfo_t *" },
797         { "nfsv4", "op-restorefh-done", 2, 1, "RESTOREFH4res *" },
798         { "nfsv4", "op-savefh-start", 0, 0, "struct compound_state *",
799           "conninfo_t *" },
800         { "nfsv4", "op-savefh-start", 1, 0, "struct compound_state *",
801           "nfsv4opinfo_t *" },
802         { "nfsv4", "op-savefh-done", 0, 0, "struct compound_state *",
803           "conninfo_t *" },
804         { "nfsv4", "op-savefh-done", 1, 0, "struct compound_state *",
805           "nfsv4opinfo_t *" },
806         { "nfsv4", "op-savefh-done", 2, 1, "SAVEFH4res *" },
807         { "nfsv4", "op-secinfo-start", 0, 0, "struct compound_state *",
808           "conninfo_t *" },
809         { "nfsv4", "op-secinfo-start", 1, 0, "struct compound_state *",
810           "nfsv4opinfo_t *" },
811         { "nfsv4", "op-secinfo-start", 2, 1, "SECINFO4args *" },
812         { "nfsv4", "op-secinfo-done", 0, 0, "struct compound_state *",
813           "conninfo_t *" },
814         { "nfsv4", "op-secinfo-done", 1, 0, "struct compound_state *",
815           "nfsv4opinfo_t *" },
816         { "nfsv4", "op-secinfo-done", 2, 1, "SECINFO4res *" },
817         { "nfsv4", "op-setattr-start", 0, 0, "struct compound_state *",
818           "conninfo_t *" },
819         { "nfsv4", "op-setattr-start", 1, 0, "struct compound_state *",
820           "nfsv4opinfo_t *" },
821         { "nfsv4", "op-setattr-start", 2, 1, "SETATTR4args *" },
822         { "nfsv4", "op-setattr-done", 0, 0, "struct compound_state *",
823           "conninfo_t *" },
824         { "nfsv4", "op-setattr-done", 1, 0, "struct compound_state *",
825           "nfsv4opinfo_t *" },
826         { "nfsv4", "op-setattr-done", 2, 1, "SETATTR4res *" },
827         { "nfsv4", "op-setclientid-start", 0, 0, "struct compound_state *",
828           "conninfo_t *" },
829         { "nfsv4", "op-setclientid-start", 1, 0, "struct compound_state *",
830           "nfsv4opinfo_t *" },
831         { "nfsv4", "op-setclientid-start", 2, 1, "SETCLIENTID4args *" },
832         { "nfsv4", "op-setclientid-done", 0, 0, "struct compound_state *",
833           "conninfo_t *" },
834         { "nfsv4", "op-setclientid-done", 1, 0, "struct compound_state *",
835           "nfsv4opinfo_t *" },
836         { "nfsv4", "op-setclientid-done", 2, 1, "SETCLIENTID4res *" },
837         { "nfsv4", "op-setclientid-confirm-start", 0, 0,
838           "struct compound_state *", "conninfo_t *" },
839         { "nfsv4", "op-setclientid-confirm-start", 1, 0,
840           "struct compound_state *", "nfsv4opinfo_t *" },
```

```
841         { "nfsv4", "op-setclientid-confirm-start", 2, 1,
842             "SETCLIENTID_CONFIRM4args *" },
843         { "nfsv4", "op-setclientid-confirm-done", 0, 0,
844             "struct compound_state *", "conninfo_t *" },
845         { "nfsv4", "op-setclientid-confirm-done", 1, 0,
846             "struct compound_state *", "nfsv4opinfo_t *" },
847         { "nfsv4", "op-setclientid-confirm-done", 2, 1,
848             "SETCLIENTID_CONFIRM4res *" },
849         { "nfsv4", "op-verify-start", 0, 0, "struct compound_state *",
850             "conninfo_t *" },
851         { "nfsv4", "op-verify-start", 1, 0, "struct compound_state *",
852             "nfsv4opinfo_t *" },
853         { "nfsv4", "op-verify-start", 2, 1, "VERIFY4args *" },
854         { "nfsv4", "op-verify-done", 0, 0, "struct compound_state *",
855             "conninfo_t *" },
856         { "nfsv4", "op-verify-done", 1, 0, "struct compound_state *",
857             "nfsv4opinfo_t *" },
858         { "nfsv4", "op-verify-done", 2, 1, "VERIFY4res *" },
859         { "nfsv4", "op-write-start", 0, 0, "struct compound_state *",
860             "conninfo_t *" },
861         { "nfsv4", "op-write-start", 1, 0, "struct compound_state *",
862             "nfsv4opinfo_t *" },
863         { "nfsv4", "op-write-start", 2, 1, "WRITE4args *" },
864         { "nfsv4", "op-write-done", 0, 0, "struct compound_state *",
865             "conninfo_t *" },
866         { "nfsv4", "op-write-done", 1, 0, "struct compound_state *",
867             "nfsv4opinfo_t *" },
868         { "nfsv4", "op-write-done", 2, 1, "WRITE4res *" },
869         { "nfsv4", "cb-recall-start", 0, 0, "rfs4_client_t *",
870             "conninfo_t *" },
871         { "nfsv4", "cb-recall-start", 1, 1, "rfs4_deleg_state_t *",
872             "nfsv4cbinfo_t *" },
873         { "nfsv4", "cb-recall-start", 2, 2, "CB_RECALL4args *" },
874         { "nfsv4", "cb-recall-done", 0, 0, "rfs4_client_t *",
875             "conninfo_t *" },
876         { "nfsv4", "cb-recall-done", 1, 1, "rfs4_deleg_state_t *",
877             "nfsv4cbinfo_t *" },
878         { "nfsv4", "cb-recall-done", 2, 2, "CB_RECALL4res *" },

880         /* Tables like this get really ugly when line-wrapped. */
881         /* BEGIN CSTYLED */
882         { "smb", "op-Close-start", 0, 0, "smb_request_t *", "conninfo_t *" },
883         { "smb", "op-Close-start", 1, 0, "smb_request_t *", "smbopinfo_t *" },
884         { "smb", "op-Close-done", 0, 0, "smb_request_t *", "conninfo_t *" },
885         { "smb", "op-Close-done", 1, 0, "smb_request_t *", "smbopinfo_t *" },

887         { "smb", "op-CloseAndTreeDisconnect-start", 0, 0, "smb_request_t *", "co
888         { "smb", "op-CloseAndTreeDisconnect-start", 1, 0, "smb_request_t *", "sm
889         { "smb", "op-CloseAndTreeDisconnect-done", 0, 0, "smb_request_t *", "con
890         { "smb", "op-CloseAndTreeDisconnect-done", 1, 0, "smb_request_t *", "smb

892         { "smb", "op-Transaction-start", 0, 0, "smb_request_t *", "conninfo_t *"
893         { "smb", "op-Transaction-start", 1, 0, "smb_request_t *", "smbopinfo_t *
894         { "smb", "op-Transaction-done", 0, 0, "smb_request_t *", "conninfo_t *"
895         { "smb", "op-Transaction-done", 1, 0, "smb_request_t *", "smbopinfo_t *"

897         { "smb", "op-TransactionSecondary-start", 0, 0, "smb_request_t *", "conn
898         { "smb", "op-TransactionSecondary-start", 1, 0, "smb_request_t *", "smbo
899         { "smb", "op-TransactionSecondary-done", 0, 0, "smb_request_t *", "conni
900         { "smb", "op-TransactionSecondary-done", 1, 0, "smb_request_t *", "smbop

902         { "smb", "op-Ioctl-start", 0, 0, "smb_request_t *", "conninfo_t *" },
903         { "smb", "op-Ioctl-start", 1, 0, "smb_request_t *", "smbopinfo_t *" },
904         { "smb", "op-Ioctl-done", 0, 0, "smb_request_t *", "conninfo_t *" },
905         { "smb", "op-Ioctl-done", 1, 0, "smb_request_t *", "smbopinfo_t *" },
```



```
907         { "smb", "op-Transaction2-start", 0, 0, "smb_request_t *", "conninfo_t *
908         { "smb", "op-Transaction2-start", 1, 0, "smb_request_t *", "smbopinfo_t
909         { "smb", "op-Transaction2-done", 0, 0, "smb_request_t *", "conninfo_t *"
910         { "smb", "op-Transaction2-done", 1, 0, "smb_request_t *", "smbopinfo_t *

912         { "smb", "op-Transaction2Secondary-start", 0, 0, "smb_request_t *", "con
913         { "smb", "op-Transaction2Secondary-start", 1, 0, "smb_request_t *", "smb
914         { "smb", "op-Transaction2Secondary-done", 0, 0, "smb_request_t *", "conn
915         { "smb", "op-Transaction2Secondary-done", 1, 0, "smb_request_t *", "smbo

917         { "smb", "op-NtTransact-start", 0, 0, "smb_request_t *", "conninfo_t *"
918         { "smb", "op-NtTransact-start", 1, 0, "smb_request_t *", "smbopinfo_t *"
919         { "smb", "op-NtTransact-done", 0, 0, "smb_request_t *", "conninfo_t *" }
920         { "smb", "op-NtTransact-done", 1, 0, "smb_request_t *", "smbopinfo_t *"

922         { "smb", "op-NtTransactSecondary-start", 0, 0, "smb_request_t *", "conni
923         { "smb", "op-NtTransactSecondary-start", 1, 0, "smb_request_t *", "smbop
924         { "smb", "op-NtTransactSecondary-done", 0, 0, "smb_request_t *", "connin
925         { "smb", "op-NtTransactSecondary-done", 1, 0, "smb_request_t *", "smbopi

927         { "smb", "op-Create-start", 0, 0, "smb_request_t *", "conninfo_t *" },
928         { "smb", "op-Create-start", 1, 0, "smb_request_t *", "smbopinfo_t *" },
929         { "smb", "op-Create-start", 2, 0, "smb_request_t *", "smb_open_args_t *"
930         { "smb", "op-Create-done", 0, 0, "smb_request_t *", "conninfo_t *" },
931         { "smb", "op-Create-done", 1, 0, "smb_request_t *", "smbopinfo_t *" },

933         { "smb", "op-CreateNew-start", 0, 0, "smb_request_t *", "conninfo_t *" }
934         { "smb", "op-CreateNew-start", 1, 0, "smb_request_t *", "smbopinfo_t *"
935         { "smb", "op-CreateNew-start", 2, 0, "smb_request_t *", "smb_open_args_t
936         { "smb", "op-CreateNew-done", 0, 0, "smb_request_t *", "conninfo_t *" },
937         { "smb", "op-CreateNew-done", 1, 0, "smb_request_t *", "smbopinfo_t *" }

939         { "smb", "op-CreateTemporary-start", 0, 0, "smb_request_t *", "conninfo_
940         { "smb", "op-CreateTemporary-start", 1, 0, "smb_request_t *", "smbopinfo
941         { "smb", "op-CreateTemporary-start", 2, 0, "smb_request_t *", "smb_open_
942         { "smb", "op-CreateTemporary-done", 0, 0, "smb_request_t *", "conninfo_t
943         { "smb", "op-CreateTemporary-done", 1, 0, "smb_request_t *", "smbopinfo_

945         { "smb", "op-Delete-start", 0, 0, "smb_request_t *", "conninfo_t *" },
946         { "smb", "op-Delete-start", 1, 0, "smb_request_t *", "smbopinfo_t *" },
947         { "smb", "op-Delete-start", 2, 0, "smb_request_t *", "smb_name_args_t *"
948         { "smb", "op-Delete-done", 0, 0, "smb_request_t *", "conninfo_t *" },
949         { "smb", "op-Delete-done", 1, 0, "smb_request_t *", "smbopinfo_t *" },

951         { "smb", "op-CreateDirectory-start", 0, 0, "smb_request_t *", "conninfo_
952         { "smb", "op-CreateDirectory-start", 1, 0, "smb_request_t *", "smbopinfo
953         { "smb", "op-CreateDirectory-start", 2, 0, "smb_request_t *", "smb_name_
954         { "smb", "op-CreateDirectory-done", 0, 0, "smb_request_t *", "conninfo_t
955         { "smb", "op-CreateDirectory-done", 1, 0, "smb_request_t *", "smbopinfo_

957         { "smb", "op-DeleteDirectory-start", 0, 0, "smb_request_t *", "conninfo_
958         { "smb", "op-DeleteDirectory-start", 1, 0, "smb_request_t *", "smbopinfo
959         { "smb", "op-DeleteDirectory-start", 2, 0, "smb_request_t *", "smb_name_
960         { "smb", "op-DeleteDirectory-done", 0, 0, "smb_request_t *", "conninfo_t
961         { "smb", "op-DeleteDirectory-done", 1, 0, "smb_request_t *", "smbopinfo_

963         { "smb", "op-CheckDirectory-start", 0, 0, "smb_request_t *", "conninfo_t
964         { "smb", "op-CheckDirectory-start", 1, 0, "smb_request_t *", "smbopinfo_
965         { "smb", "op-CheckDirectory-start", 2, 0, "smb_request_t *", "smb_name_a
966         { "smb", "op-CheckDirectory-done", 0, 0, "smb_request_t *", "conninfo_t
967         { "smb", "op-CheckDirectory-done", 1, 0, "smb_request_t *", "smbopinfo_t

969         { "smb", "op-Invalid-start", 0, 0, "smb_request_t *", "conninfo_t *" },
970         { "smb", "op-Invalid-start", 1, 0, "smb_request_t *", "smbopinfo_t *" },
971         { "smb", "op-Invalid-done", 0, 0, "smb_request_t *", "conninfo_t *" },
972         { "smb", "op-Invalid-done", 1, 0, "smb_request_t *", "smbopinfo_t *" },
```

```
974         { "smb", "op-Echo-start", 0, 0, "smb_request_t *", "conninfo_t *" },
975         { "smb", "op-Echo-start", 1, 0, "smb_request_t *", "smbopinfo_t *" },
976         { "smb", "op-Echo-done", 0, 0, "smb_request_t *", "conninfo_t *" },
977         { "smb", "op-Echo-done", 1, 0, "smb_request_t *", "smbopinfo_t *" },

979         { "smb", "op-Search-start", 0, 0, "smb_request_t *", "conninfo_t *" },
980         { "smb", "op-Search-start", 1, 0, "smb_request_t *", "smbopinfo_t *" },
981         { "smb", "op-Search-done", 0, 0, "smb_request_t *", "conninfo_t *" },
982         { "smb", "op-Search-done", 1, 0, "smb_request_t *", "smbopinfo_t *" },

984         { "smb", "op-Find-start", 0, 0, "smb_request_t *", "conninfo_t *" },
985         { "smb", "op-Find-start", 1, 0, "smb_request_t *", "smbopinfo_t *" },
986         { "smb", "op-Find-done", 0, 0, "smb_request_t *", "conninfo_t *" },
987         { "smb", "op-Find-done", 1, 0, "smb_request_t *", "smbopinfo_t *" },

989         { "smb", "op-FindClose-start", 0, 0, "smb_request_t *", "conninfo_t *" }
990         { "smb", "op-FindClose-start", 1, 0, "smb_request_t *", "smbopinfo_t *"
991         { "smb", "op-FindClose-done", 0, 0, "smb_request_t *", "conninfo_t *" },
992         { "smb", "op-FindClose-done", 1, 0, "smb_request_t *", "smbopinfo_t *" }

994         { "smb", "op-FindUnique-start", 0, 0, "smb_request_t *", "conninfo_t *"
995         { "smb", "op-FindUnique-start", 1, 0, "smb_request_t *", "smbopinfo_t *"
996         { "smb", "op-FindUnique-done", 0, 0, "smb_request_t *", "conninfo_t *" }
997         { "smb", "op-FindUnique-done", 1, 0, "smb_request_t *", "smbopinfo_t *"

999         { "smb", "op-Flush-start", 0, 0, "smb_request_t *", "conninfo_t *" },
1000        { "smb", "op-Flush-start", 1, 0, "smb_request_t *", "smbopinfo_t *" },
1001        { "smb", "op-Flush-done", 0, 0, "smb_request_t *", "conninfo_t *" },
1002        { "smb", "op-Flush-done", 1, 0, "smb_request_t *", "smbopinfo_t *" },

1004        { "smb", "op-QueryInformationDisk-start", 0, 0, "smb_request_t *", "conn
1005        { "smb", "op-QueryInformationDisk-start", 1, 0, "smb_request_t *", "smbo
1006        { "smb", "op-QueryInformationDisk-done", 0, 0, "smb_request_t *", "conni
1007        { "smb", "op-QueryInformationDisk-done", 1, 0, "smb_request_t *", "smbop

1009        { "smb", "op-LockByteRange-start", 0, 0, "smb_request_t *", "conninfo_t
1010        { "smb", "op-LockByteRange-start", 1, 0, "smb_request_t *", "smbopinfo_t
1011        { "smb", "op-LockByteRange-done", 0, 0, "smb_request_t *", "conninfo_t *
1012        { "smb", "op-LockByteRange-done", 1, 0, "smb_request_t *", "smbopinfo_t

1014        { "smb", "op-LockingX-start", 0, 0, "smb_request_t *", "conninfo_t *" },
1015        { "smb", "op-LockingX-start", 1, 0, "smb_request_t *", "smbopinfo_t *" }
1016        { "smb", "op-LockingX-done", 0, 0, "smb_request_t *", "conninfo_t *" },
1017        { "smb", "op-LockingX-done", 1, 0, "smb_request_t *", "smbopinfo_t *" },

1019        { "smb", "op-LogoffX-start", 0, 0, "smb_request_t *", "conninfo_t *" },
1020        { "smb", "op-LogoffX-start", 1, 0, "smb_request_t *", "smbopinfo_t *" },
1021        { "smb", "op-LogoffX-done", 0, 0, "smb_request_t *", "conninfo_t *" },
1022        { "smb", "op-LogoffX-done", 1, 0, "smb_request_t *", "smbopinfo_t *" },

1024        { "smb", "op-Negotiate-start", 0, 0, "smb_request_t *", "conninfo_t *" }
1025        { "smb", "op-Negotiate-start", 1, 0, "smb_request_t *", "smbopinfo_t *"
1026        { "smb", "op-Negotiate-done", 0, 0, "smb_request_t *", "conninfo_t *" },
1027        { "smb", "op-Negotiate-done", 1, 0, "smb_request_t *", "smbopinfo_t *" }

1029        { "smb", "op-NtCancel-start", 0, 0, "smb_request_t *", "conninfo_t *" },
1030        { "smb", "op-NtCancel-start", 1, 0, "smb_request_t *", "smbopinfo_t *" }
1031        { "smb", "op-NtCancel-done", 0, 0, "smb_request_t *", "conninfo_t *" },
1032        { "smb", "op-NtCancel-done", 1, 0, "smb_request_t *", "smbopinfo_t *" },

1034        { "smb", "op-NtCreateX-start", 0, 0, "smb_request_t *", "conninfo_t *" }
1035        { "smb", "op-NtCreateX-start", 1, 0, "smb_request_t *", "smbopinfo_t *"
1036        { "smb", "op-NtCreateX-start", 2, 0, "smb_request_t *", "smb_open_args_t
1037        { "smb", "op-NtCreateX-done", 0, 0, "smb_request_t *", "conninfo_t *" },
1038        { "smb", "op-NtCreateX-done", 1, 0, "smb_request_t *", "smbopinfo_t *" }
```



```
1040        { "smb", "op-NtTransactCreate-start", 0, 0, "smb_request_t *", "conninfo
1041        { "smb", "op-NtTransactCreate-start", 1, 0, "smb_request_t *", "smbopinf
1042        { "smb", "op-NtTransactCreate-start", 2, 0, "smb_request_t *", "smb_open
1043        { "smb", "op-NtTransactCreate-done", 0, 0, "smb_request_t *", "conninfo_
1044        { "smb", "op-NtTransactCreate-done", 1, 0, "smb_request_t *", "smbopinfo

1046        { "smb", "op-Open-start", 0, 0, "smb_request_t *", "conninfo_t *" },
1047        { "smb", "op-Open-start", 1, 0, "smb_request_t *", "smbopinfo_t *" },
1048        { "smb", "op-Open-start", 2, 0, "smb_request_t *", "smb_open_args_t *" }
1049        { "smb", "op-Open-done", 0, 0, "smb_request_t *", "conninfo_t *" },
1050        { "smb", "op-Open-done", 1, 0, "smb_request_t *", "smbopinfo_t *" },

1052        { "smb", "op-OpenX-start", 0, 0, "smb_request_t *", "conninfo_t *" },
1053        { "smb", "op-OpenX-start", 1, 0, "smb_request_t *", "smbopinfo_t *" },
1054        { "smb", "op-OpenX-start", 2, 0, "smb_request_t *", "smb_open_args_t *"
1055        { "smb", "op-OpenX-done", 0, 0, "smb_request_t *", "conninfo_t *" },
1056        { "smb", "op-OpenX-done", 1, 0, "smb_request_t *", "smbopinfo_t *" },

1058        { "smb", "op-OpenPrintFile-start", 0, 0, "smb_request_t *", "conninfo_t
1059        { "smb", "op-OpenPrintFile-start", 1, 0, "smb_request_t *", "smbopinfo_t
1060        { "smb", "op-OpenPrintFile-start", 2, 0, "smb_request_t *", "smb_open_ar
1061        { "smb", "op-OpenPrintFile-done", 0, 0, "smb_request_t *", "conninfo_t *
1062        { "smb", "op-OpenPrintFile-done", 1, 0, "smb_request_t *", "smbopinfo_t

1064        { "smb", "op-ClosePrintFile-start", 0, 0, "smb_request_t *", "conninfo_t
1065        { "smb", "op-ClosePrintFile-start", 1, 0, "smb_request_t *", "smbopinfo_
1066        { "smb", "op-ClosePrintFile-done", 0, 0, "smb_request_t *", "conninfo_t
1067        { "smb", "op-ClosePrintFile-done", 1, 0, "smb_request_t *", "smbopinfo_t

1069        { "smb", "op-GetPrintQueue-start", 0, 0, "smb_request_t *", "conninfo_t
1070        { "smb", "op-GetPrintQueue-start", 1, 0, "smb_request_t *", "smbopinfo_t
1071        { "smb", "op-GetPrintQueue-done", 0, 0, "smb_request_t *", "conninfo_t *
1072        { "smb", "op-GetPrintQueue-done", 1, 0, "smb_request_t *", "smbopinfo_t

1074        { "smb", "op-WritePrintFile-start", 0, 0, "smb_request_t *", "conninfo_t
1075        { "smb", "op-WritePrintFile-start", 1, 0, "smb_request_t *", "smbopinfo_
1076        { "smb", "op-WritePrintFile-done", 0, 0, "smb_request_t *", "conninfo_t
1077        { "smb", "op-WritePrintFile-done", 1, 0, "smb_request_t *", "smbopinfo_t

1079        { "smb", "op-ProcessExit-start", 0, 0, "smb_request_t *", "conninfo_t *"
1080        { "smb", "op-ProcessExit-start", 1, 0, "smb_request_t *", "smbopinfo_t *
1081        { "smb", "op-ProcessExit-done", 0, 0, "smb_request_t *", "conninfo_t *"
1082        { "smb", "op-ProcessExit-done", 1, 0, "smb_request_t *", "smbopinfo_t *"

1084        { "smb", "op-QueryInformation-start", 0, 0, "smb_request_t *", "conninfo
1085        { "smb", "op-QueryInformation-start", 1, 0, "smb_request_t *", "smbopinf
1086        { "smb", "op-QueryInformation-start", 2, 0, "smb_request_t *", "smb_name
1087        { "smb", "op-QueryInformation-done", 0, 0, "smb_request_t *", "conninfo_
1088        { "smb", "op-QueryInformation-done", 1, 0, "smb_request_t *", "smbopinfo

1090        { "smb", "op-QueryInformation2-start", 0, 0, "smb_request_t *", "conninf
1091        { "smb", "op-QueryInformation2-start", 1, 0, "smb_request_t *", "smbopin
1092        { "smb", "op-QueryInformation2-done", 0, 0, "smb_request_t *", "conninfo
1093        { "smb", "op-QueryInformation2-done", 1, 0, "smb_request_t *", "smbopinf

1095        { "smb", "op-Read-start", 0, 0, "smb_request_t *", "conninfo_t *" },
1096        { "smb", "op-Read-start", 1, 0, "smb_request_t *", "smbopinfo_t *" },
1097        { "smb", "op-Read-start", 2, 0, "smb_request_t *", "smb_rw_args_t *" },
1098        { "smb", "op-Read-done", 0, 0, "smb_request_t *", "conninfo_t *" },
1099        { "smb", "op-Read-done", 1, 0, "smb_request_t *", "smbopinfo_t *" },
1100        { "smb", "op-Read-done", 2, 0, "smb_request_t *", "smb_rw_args_t *" },

1102        { "smb", "op-LockAndRead-start", 0, 0, "smb_request_t *", "conninfo_t *"
1103        { "smb", "op-LockAndRead-start", 1, 0, "smb_request_t *", "smbopinfo_t *
1104        { "smb", "op-LockAndRead-start", 2, 0, "smb_request_t *", "smb_rw_args_t
```

```
1105         { "smb", "op-LockAndRead-done", 0, 0, "smb_request_t *", "conninfo_t *"
1106         { "smb", "op-LockAndRead-done", 1, 0, "smb_request_t *", "smbopinfo_t *"
1107         { "smb", "op-LockAndRead-done", 2, 0, "smb_request_t *", "smb_rw_args_t

1109         { "smb", "op-ReadRaw-start", 0, 0, "smb_request_t *", "conninfo_t *" },
1110         { "smb", "op-ReadRaw-start", 1, 0, "smb_request_t *", "smbopinfo_t *" },
1111         { "smb", "op-ReadRaw-start", 2, 0, "smb_request_t *", "smb_rw_args_t *"
1112         { "smb", "op-ReadRaw-done", 0, 0, "smb_request_t *", "conninfo_t *" },
1113         { "smb", "op-ReadRaw-done", 1, 0, "smb_request_t *", "smbopinfo_t *" },
1114         { "smb", "op-ReadRaw-done", 2, 0, "smb_request_t *", "smb_rw_args_t *" }

1116         { "smb", "op-ReadX-start", 0, 0, "smb_request_t *", "conninfo_t *" },
1117         { "smb", "op-ReadX-start", 1, 0, "smb_request_t *", "smbopinfo_t *" },
1118         { "smb", "op-ReadX-start", 2, 0, "smb_request_t *", "smb_rw_args_t *" },
1119         { "smb", "op-ReadX-done", 0, 0, "smb_request_t *", "conninfo_t *" },
1120         { "smb", "op-ReadX-done", 1, 0, "smb_request_t *", "smbopinfo_t *" },
1121         { "smb", "op-ReadX-done", 2, 0, "smb_request_t *", "smb_rw_args_t *" },

1123         { "smb", "op-Rename-start", 0, 0, "smb_request_t *", "conninfo_t *" },
1124         { "smb", "op-Rename-start", 1, 0, "smb_request_t *", "smbopinfo_t *" },
1125         { "smb", "op-Rename-start", 2, 0, "smb_request_t *", "smb_name_args_t *"
1126         { "smb", "op-Rename-done", 0, 0, "smb_request_t *", "conninfo_t *" },
1127         { "smb", "op-Rename-done", 1, 0, "smb_request_t *", "smbopinfo_t *" },

1129         { "smb", "op-NtRename-start", 0, 0, "smb_request_t *", "conninfo_t *" },
1130         { "smb", "op-NtRename-start", 1, 0, "smb_request_t *", "smbopinfo_t *" }
1131         { "smb", "op-NtRename-start", 2, 0, "smb_request_t *", "smb_name_args_t
1132         { "smb", "op-NtRename-done", 0, 0, "smb_request_t *", "conninfo_t *" },
1133         { "smb", "op-NtRename-done", 1, 0, "smb_request_t *", "smbopinfo_t *" },

1135         { "smb", "op-Seek-start", 0, 0, "smb_request_t *", "conninfo_t *" },
1136         { "smb", "op-Seek-start", 1, 0, "smb_request_t *", "smbopinfo_t *" },
1137         { "smb", "op-Seek-done", 0, 0, "smb_request_t *", "conninfo_t *" },
1138         { "smb", "op-Seek-done", 1, 0, "smb_request_t *", "smbopinfo_t *" },

1140         { "smb", "op-SessionSetupX-start", 0, 0, "smb_request_t *", "conninfo_t
1141         { "smb", "op-SessionSetupX-start", 1, 0, "smb_request_t *", "smbopinfo_t
1142         { "smb", "op-SessionSetupX-done", 0, 0, "smb_request_t *", "conninfo_t *
1143         { "smb", "op-SessionSetupX-done", 1, 0, "smb_request_t *", "smbopinfo_t

1145         { "smb", "op-SetInformation-start", 0, 0, "smb_request_t *", "conninfo_t
1146         { "smb", "op-SetInformation-start", 1, 0, "smb_request_t *", "smbopinfo_
1147         { "smb", "op-SetInformation-done", 0, 0, "smb_request_t *", "conninfo_t
1148         { "smb", "op-SetInformation-done", 1, 0, "smb_request_t *", "smbopinfo_t

1150         { "smb", "op-SetInformation2-start", 0, 0, "smb_request_t *", "conninfo_
1151         { "smb", "op-SetInformation2-start", 1, 0, "smb_request_t *", "smbopinfo
1152         { "smb", "op-SetInformation2-done", 0, 0, "smb_request_t *", "conninfo_t
1153         { "smb", "op-SetInformation2-done", 1, 0, "smb_request_t *", "smbopinfo_

1155         { "smb", "op-FindClose2-start", 0, 0, "smb_request_t *", "conninfo_t *"
1156         { "smb", "op-FindClose2-start", 1, 0, "smb_request_t *", "smbopinfo_t *"
1157         { "smb", "op-FindClose2-done", 0, 0, "smb_request_t *", "conninfo_t *" }
1158         { "smb", "op-FindClose2-done", 1, 0, "smb_request_t *", "smbopinfo_t *"

1160         { "smb", "op-TreeConnect-start", 0, 0, "smb_request_t *", "conninfo_t *"
1161         { "smb", "op-TreeConnect-start", 1, 0, "smb_request_t *", "smbopinfo_t *
1162         { "smb", "op-TreeConnect-done", 0, 0, "smb_request_t *", "conninfo_t *"
1163         { "smb", "op-TreeConnect-done", 1, 0, "smb_request_t *", "smbopinfo_t *"

1165         { "smb", "op-TreeConnectX-start", 0, 0, "smb_request_t *", "conninfo_t *
1166         { "smb", "op-TreeConnectX-start", 1, 0, "smb_request_t *", "smbopinfo_t
1167         { "smb", "op-TreeConnectX-done", 0, 0, "smb_request_t *", "conninfo_t *"
1168         { "smb", "op-TreeConnectX-done", 1, 0, "smb_request_t *", "smbopinfo_t *

1170         { "smb", "op-TreeDisconnect-start", 0, 0, "smb_request_t *", "conninfo_t
```



```
1171         { "smb", "op-TreeDisconnect-start", 1, 0, "smb_request_t *", "smbopinfo_
1172         { "smb", "op-TreeDisconnect-done", 0, 0, "smb_request_t *", "conninfo_t
1173         { "smb", "op-TreeDisconnect-done", 1, 0, "smb_request_t *", "smbopinfo_t

1175         { "smb", "op-UnlockByteRange-start", 0, 0, "smb_request_t *", "conninfo_
1176         { "smb", "op-UnlockByteRange-start", 1, 0, "smb_request_t *", "smbopinfo
1177         { "smb", "op-UnlockByteRange-done", 0, 0, "smb_request_t *", "conninfo_t
1178         { "smb", "op-UnlockByteRange-done", 1, 0, "smb_request_t *", "smbopinfo_

1180         { "smb", "op-Write-start", 0, 0, "smb_request_t *", "conninfo_t *" },
1181         { "smb", "op-Write-start", 1, 0, "smb_request_t *", "smbopinfo_t *" },
1182         { "smb", "op-Write-start", 2, 0, "smb_request_t *", "smb_rw_args_t *" },
1183         { "smb", "op-Write-done", 0, 0, "smb_request_t *", "conninfo_t *" },
1184         { "smb", "op-Write-done", 1, 0, "smb_request_t *", "smbopinfo_t *" },
1185         { "smb", "op-Write-done", 2, 0, "smb_request_t *", "smb_rw_args_t *" },

1187         { "smb", "op-WriteAndClose-start", 0, 0, "smb_request_t *", "conninfo_t
1188         { "smb", "op-WriteAndClose-start", 1, 0, "smb_request_t *", "smbopinfo_t
1189         { "smb", "op-WriteAndClose-start", 2, 0, "smb_request_t *", "smb_rw_args
1190         { "smb", "op-WriteAndClose-done", 0, 0, "smb_request_t *", "conninfo_t *
1191         { "smb", "op-WriteAndClose-done", 1, 0, "smb_request_t *", "smbopinfo_t
1192         { "smb", "op-WriteAndClose-done", 2, 0, "smb_request_t *", "smb_rw_args_

1194         { "smb", "op-WriteAndUnlock-start", 0, 0, "smb_request_t *", "conninfo_t
1195         { "smb", "op-WriteAndUnlock-start", 1, 0, "smb_request_t *", "smbopinfo_
1196         { "smb", "op-WriteAndUnlock-start", 2, 0, "smb_request_t *", "smb_rw_arg
1197         { "smb", "op-WriteAndUnlock-done", 0, 0, "smb_request_t *", "conninfo_t
1198         { "smb", "op-WriteAndUnlock-done", 1, 0, "smb_request_t *", "smbopinfo_t
1199         { "smb", "op-WriteAndUnlock-done", 2, 0, "smb_request_t *", "smb_rw_args

1201         { "smb", "op-WriteRaw-start", 0, 0, "smb_request_t *", "conninfo_t *" },
1202         { "smb", "op-WriteRaw-start", 1, 0, "smb_request_t *", "smbopinfo_t *" }
1203         { "smb", "op-WriteRaw-start", 2, 0, "smb_request_t *", "smb_rw_args_t *"
1204         { "smb", "op-WriteRaw-done", 0, 0, "smb_request_t *", "conninfo_t *" },
1205         { "smb", "op-WriteRaw-done", 1, 0, "smb_request_t *", "smbopinfo_t *" },
1206         { "smb", "op-WriteRaw-done", 2, 0, "smb_request_t *", "smb_rw_args_t *"

1208         { "smb", "op-WriteX-start", 0, 0, "smb_request_t *", "conninfo_t *" },
1209         { "smb", "op-WriteX-start", 1, 0, "smb_request_t *", "smbopinfo_t *" },
1210         { "smb", "op-WriteX-start", 2, 0, "smb_request_t *", "smb_rw_args_t *" }
1211         { "smb", "op-WriteX-done", 0, 0, "smb_request_t *", "conninfo_t *" },
1212         { "smb", "op-WriteX-done", 1, 0, "smb_request_t *", "smbopinfo_t *" },
1213         { "smb", "op-WriteX-done", 2, 0, "smb_request_t *", "smb_rw_args_t *" },
1214         /* END CSTYLED */

1216         { "ip", "send", 0, 0, "mblk_t *", "pktinfo_t *" },
1217         { "ip", "send", 1, 1, "conn_t *", "csinfo_t *" },
1218         { "ip", "send", 2, 2, "void_ip_t *", "ipinfo_t *" },
1219         { "ip", "send", 3, 3, "__dtrace_ipsr_ill_t *", "ifinfo_t *" },
1220         { "ip", "send", 4, 4, "ipha_t *", "ipv4info_t *" },
1221         { "ip", "send", 5, 5, "ip6_t *", "ipv6info_t *" },
1222         { "ip", "send", 6, 6, "int" }, /* used by __dtrace_ipsr_ill_t */
1223         { "ip", "receive", 0, 0, "mblk_t *", "pktinfo_t *" },
1224         { "ip", "receive", 1, 1, "conn_t *", "csinfo_t *" },
1225         { "ip", "receive", 2, 2, "void_ip_t *", "ipinfo_t *" },
1226         { "ip", "receive", 3, 3, "__dtrace_ipsr_ill_t *", "ifinfo_t *" },
1227         { "ip", "receive", 4, 4, "ipha_t *", "ipv4info_t *" },
1228         { "ip", "receive", 5, 5, "ip6_t *", "ipv6info_t *" },
1229         { "ip", "receive", 6, 6, "int" }, /* used by __dtrace_ipsr_ill_t */

1231         { "tcp", "connect-established", 0, 0, "mblk_t *", "pktinfo_t *" },
1232         { "tcp", "connect-established", 1, 1, "ip_xmit_attr_t *",
1233             "csinfo_t *" },
1234         { "tcp", "connect-established", 2, 2, "void_ip_t *", "ipinfo_t *" },
1235         { "tcp", "connect-established", 3, 3, "tcp_t *", "tcpsinfo_t *" },
1236         { "tcp", "connect-established", 4, 4, "tcph_t *", "tcpinfo_t *" },
```

```
1237         { "tcp", "connect-refused", 0, 0, "mblk_t *", "pktinfo_t *" },
1238         { "tcp", "connect-refused", 1, 1, "ip_xmit_attr_t *", "csinfo_t *" },
1239         { "tcp", "connect-refused", 2, 2, "void_ip_t *", "ipinfo_t *" },
1240         { "tcp", "connect-refused", 3, 3, "tcp_t *", "tcpsinfo_t *" },
1241         { "tcp", "connect-refused", 4, 4, "tcph_t *", "tcpinfo_t *" },
1242         { "tcp", "connect-request", 0, 0, "mblk_t *", "pktinfo_t *" },
1243         { "tcp", "connect-request", 1, 1, "ip_xmit_attr_t *", "csinfo_t *" },
1244         { "tcp", "connect-request", 2, 2, "void_ip_t *", "ipinfo_t *" },
1245         { "tcp", "connect-request", 3, 3, "tcp_t *", "tcpsinfo_t *" },
1246         { "tcp", "connect-request", 4, 4, "tcph_t *", "tcpinfo_t *" },
1247         { "tcp", "accept-established", 0, 0, "mblk_t *", "pktinfo_t *" },
1248         { "tcp", "accept-established", 1, 1, "ip_xmit_attr_t *", "csinfo_t *" },
1249         { "tcp", "accept-established", 2, 2, "void_ip_t *", "ipinfo_t *" },
1250         { "tcp", "accept-established", 3, 3, "tcp_t *", "tcpsinfo_t *" },
1251         { "tcp", "accept-established", 4, 4, "tcph_t *", "tcpinfo_t *" },
1252         { "tcp", "accept-refused", 0, 0, "mblk_t *", "pktinfo_t *" },
1253         { "tcp", "accept-refused", 1, 1, "ip_xmit_attr_t *", "csinfo_t *" },
1254         { "tcp", "accept-refused", 2, 2, "void_ip_t *", "ipinfo_t *" },
1255         { "tcp", "accept-refused", 3, 3, "tcp_t *", "tcpsinfo_t *" },
1256         { "tcp", "accept-refused", 4, 4, "tcph_t *", "tcpinfo_t *" },
1257         { "tcp", "state-change", 0, 0, "void", "void" },
1258         { "tcp", "state-change", 1, 1, "ip_xmit_attr_t *", "csinfo_t *" },
1259         { "tcp", "state-change", 2, 2, "void", "void" },
1260         { "tcp", "state-change", 3, 3, "tcp_t *", "tcpsinfo_t *" },
1261         { "tcp", "state-change", 4, 4, "void", "void" },
1262         { "tcp", "state-change", 5, 5, "int32_t", "tcplsinfo_t *" },
1263         { "tcp", "send", 0, 0, "mblk_t *", "pktinfo_t *" },
1264         { "tcp", "send", 1, 1, "ip_xmit_attr_t *", "csinfo_t *" },
1265         { "tcp", "send", 2, 2, "__dtrace_tcp_void_ip_t *", "ipinfo_t *" },
1266         { "tcp", "send", 3, 3, "tcp_t *", "tcpsinfo_t *" },
1267         { "tcp", "send", 4, 4, "__dtrace_tcp_tcph_t *", "tcpinfo_t *" },
1268         { "tcp", "receive", 0, 0, "mblk_t *", "pktinfo_t *" },
1269         { "tcp", "receive", 1, 1, "ip_xmit_attr_t *", "csinfo_t *" },
1270         { "tcp", "receive", 2, 2, "__dtrace_tcp_void_ip_t *", "ipinfo_t *" },
1271         { "tcp", "receive", 3, 3, "tcp_t *", "tcpsinfo_t *" },
1272         { "tcp", "receive", 4, 4, "__dtrace_tcp_tcph_t *", "tcpinfo_t *" },

1274         { "udp", "send", 0, 0, "mblk_t *", "pktinfo_t *" },
1275         { "udp", "send", 1, 1, "ip_xmit_attr_t *", "csinfo_t *" },
1276         { "udp", "send", 2, 2, "void_ip_t *", "ipinfo_t *" },
1277         { "udp", "send", 3, 3, "udp_t *", "udpsinfo_t *" },
1278         { "udp", "send", 4, 4, "udpha_t *", "udpinfo_t *" },
1279         { "udp", "receive", 0, 0, "mblk_t *", "pktinfo_t *" },
1280         { "udp", "receive", 1, 1, "ip_xmit_attr_t *", "csinfo_t *" },
1281         { "udp", "receive", 2, 2, "void_ip_t *", "ipinfo_t *" },
1282         { "udp", "receive", 3, 3, "udp_t *", "udpsinfo_t *" },
1283         { "udp", "receive", 4, 4, "udpha_t *", "udpinfo_t *" },

1285         { "sysevent", "post", 0, 0, "evch_bind_t *", "syseventchaninfo_t *" },
1286         { "sysevent", "post", 1, 1, "sysevent_impl_t *", "syseventinfo_t *" },

1288         { "xpv", "add-to-physmap-end", 0, 0, "int" },
1289         { "xpv", "add-to-physmap-start", 0, 0, "domid_t" },
1290         { "xpv", "add-to-physmap-start", 1, 1, "uint_t" },
1291         { "xpv", "add-to-physmap-start", 2, 2, "ulong_t" },
1292         { "xpv", "add-to-physmap-start", 3, 3, "ulong_t" },
1293         { "xpv", "decrease-reservation-end", 0, 0, "int" },
1294         { "xpv", "decrease-reservation-start", 0, 0, "domid_t" },
1295         { "xpv", "decrease-reservation-start", 1, 1, "ulong_t" },
1296         { "xpv", "decrease-reservation-start", 2, 2, "uint_t" },
1297         { "xpv", "decrease-reservation-start", 3, 3, "ulong_t *" },
1298         { "xpv", "dom-create-start", 0, 0, "xen_domctl_t *" },
1299         { "xpv", "dom-destroy-start", 0, 0, "domid_t" },
1300         { "xpv", "dom-pause-start", 0, 0, "domid_t" },
1301         { "xpv", "dom-unpause-start", 0, 0, "domid_t" },
1302         { "xpv", "dom-create-end", 0, 0, "int" },
```



```
1303         { "xpv", "dom-destroy-end", 0, 0, "int" },
1304         { "xpv", "dom-pause-end", 0, 0, "int" },
1305         { "xpv", "dom-unpause-end", 0, 0, "int" },
1306         { "xpv", "evtchn-op-end", 0, 0, "int" },
1307         { "xpv", "evtchn-op-start", 0, 0, "int" },
1308         { "xpv", "evtchn-op-start", 1, 1, "void *" },
1309         { "xpv", "increase-reservation-end", 0, 0, "int" },
1310         { "xpv", "increase-reservation-start", 0, 0, "domid_t" },
1311         { "xpv", "increase-reservation-start", 1, 1, "ulong_t" },
1312         { "xpv", "increase-reservation-start", 2, 2, "uint_t" },
1313         { "xpv", "increase-reservation-start", 3, 3, "ulong_t *" },
1314         { "xpv", "mmap-end", 0, 0, "int" },
1315         { "xpv", "mmap-entry", 0, 0, "ulong_t" },
1316         { "xpv", "mmap-entry", 1, 1, "ulong_t" },
1317         { "xpv", "mmap-entry", 2, 2, "ulong_t" },
1318         { "xpv", "mmap-start", 0, 0, "domid_t" },
1319         { "xpv", "mmap-start", 1, 1, "int" },
1320         { "xpv", "mmap-start", 2, 2, "privcmd_mmap_entry_t *" },
1321         { "xpv", "mmapbatch-end", 0, 0, "int" },
1322         { "xpv", "mmapbatch-end", 1, 1, "struct seg *" },
1323         { "xpv", "mmapbatch-end", 2, 2, "caddr_t" },
1324         { "xpv", "mmapbatch-start", 0, 0, "domid_t" },
1325         { "xpv", "mmapbatch-start", 1, 1, "int" },
1326         { "xpv", "mmapbatch-start", 2, 2, "caddr_t" },
1327         { "xpv", "mmu-ext-op-end", 0, 0, "int" },
1328         { "xpv", "mmu-ext-op-start", 0, 0, "int" },
1329         { "xpv", "mmu-ext-op-start", 1, 1, "struct mmuext_op *" },
1330         { "xpv", "mmu-update-start", 0, 0, "int" },
1331         { "xpv", "mmu-update-start", 1, 1, "int" },
1332         { "xpv", "mmu-update-start", 2, 2, "mmu_update_t *" },
1333         { "xpv", "mmu-update-end", 0, 0, "int" },
1334         { "xpv", "populate-physmap-end", 0, 0, "int" },
1335         { "xpv", "populate-physmap-start", 0, 0, "domid_t" },
1336         { "xpv", "populate-physmap-start", 1, 1, "ulong_t" },
1337         { "xpv", "populate-physmap-start", 2, 2, "ulong_t *" },
1338         { "xpv", "set-memory-map-end", 0, 0, "int" },
1339         { "xpv", "set-memory-map-start", 0, 0, "domid_t" },
1340         { "xpv", "set-memory-map-start", 1, 1, "int" },
1341         { "xpv", "set-memory-map-start", 2, 2, "struct xen_memory_map *" },
1342         { "xpv", "setvcpucontext-end", 0, 0, "int" },
1343         { "xpv", "setvcpucontext-start", 0, 0, "domid_t" },
1344         { "xpv", "setvcpucontext-start", 1, 1, "vcpu_guest_context_t *" },

1346         { "srp", "service-up", 0, 0, "srpt_session_t *", "conninfo_t *" },
1347         { "srp", "service-up", 1, 0, "srpt_session_t *", "srp_portinfo_t *" },
1348         { "srp", "service-down", 0, 0, "srpt_session_t *", "conninfo_t *" },
1349         { "srp", "service-down", 1, 0, "srpt_session_t *",
1350           "srp_portinfo_t *" },
1351         { "srp", "login-command", 0, 0, "srpt_session_t *", "conninfo_t *" },
1352         { "srp", "login-command", 1, 0, "srpt_session_t *",
1353           "srp_portinfo_t *" },
1354         { "srp", "login-command", 2, 1, "srp_login_req_t *",
1355           "srp_logininfo_t *" },
1356         { "srp", "login-response", 0, 0, "srpt_session_t *", "conninfo_t *" },
1357         { "srp", "login-response", 1, 0, "srpt_session_t *",
1358           "srp_portinfo_t *" },
1359         { "srp", "login-response", 2, 1, "srp_login_rsp_t *",
1360           "srp_logininfo_t *" },
1361         { "srp", "login-response", 3, 2, "srp_login_rej_t *" },
1362         { "srp", "logout-command", 0, 0, "srpt_channel_t *", "conninfo_t *" },
1363         { "srp", "logout-command", 1, 0, "srpt_channel_t *",
1364           "srp_portinfo_t *" },
1365         { "srp", "task-command", 0, 0, "srpt_channel_t *", "conninfo_t *" },
1366         { "srp", "task-command", 1, 0, "srpt_channel_t *",
1367           "srp_portinfo_t *" },
1368         { "srp", "task-command", 2, 1, "srp_cmd_req_t *", "srp_taskinfo_t *" },
```

```
1369         { "srp", "task-response", 0, 0, "srpt_channel_t *", "conninfo_t *" },
1370         { "srp", "task-response", 1, 0, "srpt_channel_t *",
1371           "srp_portinfo_t *" },
1372         { "srp", "task-response", 2, 1, "srp_rsp_t *", "srp_taskinfo_t *" },
1373         { "srp", "task-response", 3, 2, "scsi_task_t *" },
1374         { "srp", "task-response", 4, 3, "int8_t" },
1375         { "srp", "scsi-command", 0, 0, "srpt_channel_t *", "conninfo_t *" },
1376         { "srp", "scsi-command", 1, 0, "srpt_channel_t *",
1377           "srp_portinfo_t *" },
1378         { "srp", "scsi-command", 2, 1, "scsi_task_t *", "scsicmd_t *" },
1379         { "srp", "scsi-command", 3, 2, "srp_cmd_req_t *", "srp_taskinfo_t *" },
1380         { "srp", "scsi-response", 0, 0, "srpt_channel_t *", "conninfo_t *" },
1381         { "srp", "scsi-response", 1, 0, "srpt_channel_t *",
1382           "srp_portinfo_t *" },
1383         { "srp", "scsi-response", 2, 1, "srp_rsp_t *", "srp_taskinfo_t *" },
1384         { "srp", "scsi-response", 3, 2, "scsi_task_t *" },
1385         { "srp", "scsi-response", 4, 3, "int8_t" },
1386         { "srp", "xfer-start", 0, 0, "srpt_channel_t *", "conninfo_t *" },
1387         { "srp", "xfer-start", 1, 0, "srpt_channel_t *",
1388           "srp_portinfo_t *" },
1389         { "srp", "xfer-start", 2, 1, "ibt_wr_ds_t *", "xferinfo_t *" },
1390         { "srp", "xfer-start", 3, 2, "srpt_iu_t *", "srp_taskinfo_t *" },
1391         { "srp", "xfer-start", 4, 3, "ibt_send_wr_t *"},
1392         { "srp", "xfer-start", 5, 4, "uint32_t" },
1393         { "srp", "xfer-start", 6, 5, "uint32_t" },
1394         { "srp", "xfer-start", 7, 6, "uint32_t" },
1395         { "srp", "xfer-start", 8, 7, "uint32_t" },
1396         { "srp", "xfer-done", 0, 0, "srpt_channel_t *", "conninfo_t *" },
1397         { "srp", "xfer-done", 1, 0, "srpt_channel_t *",
1398           "srp_portinfo_t *" },
1399         { "srp", "xfer-done", 2, 1, "ibt_wr_ds_t *", "xferinfo_t *" },
1400         { "srp", "xfer-done", 3, 2, "srpt_iu_t *", "srp_taskinfo_t *" },
1401         { "srp", "xfer-done", 4, 3, "ibt_send_wr_t *"},
1402         { "srp", "xfer-done", 5, 4, "uint32_t" },
1403         { "srp", "xfer-done", 6, 5, "uint32_t" },
1404         { "srp", "xfer-done", 7, 6, "uint32_t" },
1405         { "srp", "xfer-done", 8, 7, "uint32_t" },

1407         { "fc", "link-up",   0, 0, "fct_i_local_port_t *", "conninfo_t *" },
1408         { "fc", "link-down", 0, 0, "fct_i_local_port_t *", "conninfo_t *" },
1409         { "fc", "fabric-login-start", 0, 0, "fct_i_local_port_t *",
1410           "conninfo_t *" },
1411         { "fc", "fabric-login-start", 1, 0, "fct_i_local_port_t *",
1412           "fc_port_info_t *" },
1413         { "fc", "fabric-login-end", 0, 0, "fct_i_local_port_t *",
1414           "conninfo_t *" },
1415         { "fc", "fabric-login-end", 1, 0, "fct_i_local_port_t *",
1416           "fc_port_info_t *" },
1417         { "fc", "rport-login-start", 0, 0, "fct_cmd_t *",
1418           "conninfo_t *" },
1419         { "fc", "rport-login-start", 1, 1, "fct_local_port_t *",
1420           "fc_port_info_t *" },
1421         { "fc", "rport-login-start", 2, 2, "fct_i_remote_port_t *",
1422           "fc_port_info_t *" },
1423         { "fc", "rport-login-start", 3, 3, "int", "int" },
1424         { "fc", "rport-login-end", 0, 0, "fct_cmd_t *",
1425           "conninfo_t *" },
1426         { "fc", "rport-login-end", 1, 1, "fct_local_port_t *",
1427           "fc_port_info_t *" },
1428         { "fc", "rport-login-end", 2, 2, "fct_i_remote_port_t *",
1429           "fc_port_info_t *" },
1430         { "fc", "rport-login-end", 3, 3, "int", "int" },
1431         { "fc", "rport-login-end", 4, 4, "int", "int" },
1432         { "fc", "rport-logout-start", 0, 0, "fct_cmd_t *",
1433           "conninfo_t *" },
1434         { "fc", "rport-logout-start", 1, 1, "fct_local_port_t *",
```



```
1435           "fc_port_info_t *" },
1436         { "fc", "rport-logout-start", 2, 2, "fct_i_remote_port_t *",
1437           "fc_port_info_t *" },
1438         { "fc", "rport-logout-start", 3, 3, "int", "int" },
1439         { "fc", "rport-logout-end", 0, 0, "fct_cmd_t *",
1440           "conninfo_t *" },
1441         { "fc", "rport-logout-end", 1, 1, "fct_local_port_t *",
1442           "fc_port_info_t *" },
1443         { "fc", "rport-logout-end", 2, 2, "fct_i_remote_port_t *",
1444           "fc_port_info_t *" },
1445         { "fc", "rport-logout-end", 3, 3, "int", "int" },
1446         { "fc", "scsi-command", 0, 0, "fct_cmd_t *",
1447           "conninfo_t *" },
1448         { "fc", "scsi-command", 1, 1, "fct_i_local_port_t *",
1449           "fc_port_info_t *" },
1450         { "fc", "scsi-command", 2, 2, "scsi_task_t *",
1451           "scsicmd_t *" },
1452         { "fc", "scsi-command", 3, 3, "fct_i_remote_port_t *",
1453           "fc_port_info_t *" },
1454         { "fc", "scsi-response", 0, 0, "fct_cmd_t *",
1455           "conninfo_t *" },
1456         { "fc", "scsi-response", 1, 1, "fct_i_local_port_t *",
1457           "fc_port_info_t *" },
1458         { "fc", "scsi-response", 2, 2, "scsi_task_t *",
1459           "scsicmd_t *" },
1460         { "fc", "scsi-response", 3, 3, "fct_i_remote_port_t *",
1461           "fc_port_info_t *" },
1462         { "fc", "xfer-start", 0, 0, "fct_cmd_t *",
1463           "conninfo_t *" },
1464         { "fc", "xfer-start", 1, 1, "fct_i_local_port_t *",
1465           "fc_port_info_t *" },
1466         { "fc", "xfer-start", 2, 2, "scsi_task_t *",
1467           "scsicmd_t *" },
1468         { "fc", "xfer-start", 3, 3, "fct_i_remote_port_t *",
1469           "fc_port_info_t *" },
1470         { "fc", "xfer-start", 4, 4, "stmf_data_buf_t *",
1471           "fc_xferinfo_t *" },
1472         { "fc", "xfer-done", 0, 0, "fct_cmd_t *",
1473           "conninfo_t *" },
1474         { "fc", "xfer-done", 1, 1, "fct_i_local_port_t *",
1475           "fc_port_info_t *" },
1476         { "fc", "xfer-done", 2, 2, "scsi_task_t *",
1477           "scsicmd_t *" },
1478         { "fc", "xfer-done", 3, 3, "fct_i_remote_port_t *",
1479           "fc_port_info_t *" },
1480         { "fc", "xfer-done", 4, 4, "stmf_data_buf_t *",
1481           "fc_xferinfo_t *" },
1482         { "fc", "rscn-receive", 0, 0, "fct_i_local_port_t *",
1483           "conninfo_t *" },
1484         { "fc", "rscn-receive", 1, 1, "int", "int"},
1485         { "fc", "abts-receive", 0, 0, "fct_cmd_t *",
1486           "conninfo_t *" },
1487         { "fc", "abts-receive", 1, 1, "fct_i_local_port_t *",
1488           "fc_port_info_t *" },
1489         { "fc", "abts-receive", 2, 2, "fct_i_remote_port_t *",
1490           "fc_port_info_t *" },


1493         { NULL }
1494 };
```

*unchanged_portion_omitted_*

```
**********************************************************
    5451 Fri Mar 31 14:29:39 2017
new/usr/src/uts/common/fs/smbsrv/smb_create.c
Build provider 3rd arg from smb_request_t
hacking...
**********************************************************
```

```
/*
 * 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) 2007, 2010, Oracle and/or its affiliates. All rights reserved.
 * Copyright 2016 Nexenta Systems, Inc.  All rights reserved.
 */

#include <smbsrv/smb_kproto.h>

#define SMB_CREATE_NAMEBUF_SZ   16

/*
 * Create a new file, or truncate an existing file to zero length,
 * open the file and return a fid.  The file is specified using a
 * fully qualified name relative to the tree.
 */
smb_sdrc_t
smb_pre_create(smb_request_t *sr)
{
        struct open_param *op = &sr->arg.open;
        int rc;

        bzero(op, sizeof (sr->arg.open));

        rc = smbsr_decode_vwv(sr, "wl", &op->dattr, &op->mtime.tv_sec);
        if (rc == 0)
                rc = smbsr_decode_data(sr, "%S", sr, &op->fqi.fq_path.pn_path);

        op->create_disposition = FILE_OVERWRITE_IF;
        op->create_options = FILE_NON_DIRECTORY_FILE;

        DTRACE_SMB_1(op__Create__start, smb_request_t *, sr); /* arg.open */
        DTRACE_SMB_2(op__Create__start, smb_request_t *, sr,
            struct open_param *, op);

        return ((rc == 0) ? SDRC_SUCCESS : SDRC_ERROR);
}
```

_____***unchanged_portion_omitted_***

```
/*
 * Create a new file and return a fid.  The file is specified using
 * a fully qualified name relative to the tree.
```



```
 */
smb_sdrc_t
smb_pre_create_new(smb_request_t *sr)
{
        struct open_param *op = &sr->arg.open;
        int rc;

        bzero(op, sizeof (sr->arg.open));

        rc = smbsr_decode_vwv(sr, "wl", &op->dattr, &op->mtime.tv_sec);
        if (rc == 0)
                rc = smbsr_decode_data(sr, "%S", sr, &op->fqi.fq_path.pn_path);

        op->create_disposition = FILE_CREATE;

        DTRACE_SMB_1(op__CreateNew__start, smb_request_t *, sr); /* arg.open */
        DTRACE_SMB_2(op__CreateNew__start, smb_request_t *, sr,
            struct open_param *, op);

        return ((rc == 0) ? SDRC_SUCCESS : SDRC_ERROR);
}
```

_____***unchanged_portion_omitted_***

```
/*
 * Create a unique file in the specified directory relative to the
 * current tree.  No attributes are specified.
 */
smb_sdrc_t
smb_pre_create_temporary(smb_request_t *sr)
{
        struct open_param *op = &sr->arg.open;
        uint16_t reserved;
        int rc;

        bzero(op, sizeof (sr->arg.open));

        rc = smbsr_decode_vwv(sr, "wl", &reserved, &op->mtime.tv_sec);
        if (rc == 0)
                rc = smbsr_decode_data(sr, "%S", sr, &op->fqi.fq_path.pn_path);

        op->create_disposition = FILE_CREATE;

        DTRACE_SMB_1(op__CreateTemporary__start, smb_request_t *, sr); /* arg.op
        DTRACE_SMB_2(op__CreateTemporary__start, smb_request_t *, sr,
            struct open_param *, op);

        return ((rc == 0) ? SDRC_SUCCESS : SDRC_ERROR);
}
```

_____***unchanged_portion_omitted_***

```
**********************************************************
   16983 Fri Mar 31 14:29:39 2017
new/usr/src/uts/common/fs/smbsrv/smb_delete.c
Build provider 3rd arg from smb_request_t
hacking...
NEX-1643 dtrace provider for smbsrv
Also illumos 1841:
 DTrace smb provider was mis-implemented, doesn't exist.
Add back handlers for read/write raw, so that
 legacy dtrace consumers can find the probes.
Kill extra arg in smb_negotiate
Fix missing "done" probe with smb_notify
Add example consumer: smb-trace.d
fix soi_pid
**********************************************************
```

```
   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) 2007, 2010, Oracle and/or its affiliates. All rights reserved.
  24  * Copyright 2013 Nexenta Systems, Inc. All rights reserved.
  25  */

  27 #include <sys/sunddi.h>
  28 #include <sys/nbmlock.h>

  30 #include <smbsrv/smb_kproto.h>
  31 #include <smbsrv/smb_fsops.h>
  32 #include <smbsrv/smbinfo.h>

  34 static int smb_delete_check_path(smb_request_t *);
  35 static int smb_delete_single_file(smb_request_t *, smb_error_t *);
  36 static int smb_delete_multiple_files(smb_request_t *, smb_error_t *);
  37 static int smb_delete_find_fname(smb_request_t *, smb_odir_t *, char *, int);
  38 static int smb_delete_check_dosattr(smb_request_t *, smb_error_t *);
  39 static int smb_delete_remove_file(smb_request_t *, smb_error_t *);

  41 static void smb_delete_error(smb_error_t *, uint32_t, uint16_t, uint16_t);

  43 /*
  44  * smb_com_delete
  45  *
  46  * The delete file message is sent to delete a data file. The appropriate
  47  * Tid and additional pathname are passed. Read only files may not be
  48  * deleted, the read-only attribute must be reset prior to file deletion.
  49  *
  50  * NT supports a hidden permission known as File Delete Child (FDC). If
  51  * the user has FullControl access to a directory, the user is permitted
```



```
  52  * to delete any object in the directory regardless of the permissions
  53  * on the object.
  54  *
  55  * Client Request                     Description
  56  * ================================== ================================
  57  * UCHAR WordCount;                   Count of parameter words = 1
  58  * USHORT SearchAttributes;
  59  * USHORT ByteCount;                  Count of data bytes; min = 2
  60  * UCHAR BufferFormat;                0x04
  61  * STRING FileName[];                 File name
  62  *
  63  * Multiple files may be deleted in response to a single request as
  64  * SMB_COM_DELETE supports wildcards
  65  *
  66  * SearchAttributes indicates the attributes that the target file(s) must
  67  * have. If the attribute is zero then only normal files are deleted. If
  68  * the system file or hidden attributes are specified then the delete is
  69  * inclusive -both the specified type(s) of files and normal files are
  70  * deleted. Attributes are described in the "Attribute Encoding" section
  71  * of this document.
  72  *
  73  * If bit0 of the Flags2 field of the SMB header is set, a pattern is
  74  * passed in, and the file has a long name, then the passed pattern  much
  75  * match the long file name for the delete to succeed. If bit0 is clear, a
  76  * pattern is passed in, and the file has a long name, then the passed
  77  * pattern must match the file's short name for the deletion to succeed.
  78  *
  79  * Server Response                    Description
  80  * ================================== ================================
  81  * UCHAR WordCount;                   Count of parameter words = 0
  82  * USHORT ByteCount;                  Count of data bytes = 0
  83  *
  84  * 4.2.10.1  Errors
  85  *
  86  * ERRDOS/ERRbadpath
  87  * ERRDOS/ERRbadfile
  88  * ERRDOS/ERRnoaccess
  89  * ERRDOS/ERRbadshare   # returned by NT for files that are already open
  90  * ERRHRD/ERRnowrite
  91  * ERRSRV/ERRaccess
  92  * ERRSRV/ERRinvdevice
  93  * ERRSRV/ERRinvid
  94  * ERRSRV/ERRbaduid
  95  */
  96 smb_sdrc_t
  97 smb_pre_delete(smb_request_t *sr)
  98 {
  99         int rc;
 100         smb_fqi_t *fqi;

 102         fqi = &sr->arg.dirop.fqi;

 104         if ((rc = smbsr_decode_vwv(sr, "w", &fqi->fq_sattr)) == 0)
 105                 rc = smbsr_decode_data(sr, "%S", sr, &fqi->fq_path.pn_path);

 107         DTRACE_SMB_1(op__Delete__start, smb_request_t *, sr); /* arg.dirop */
 107         DTRACE_SMB_2(op__Delete__start, smb_request_t *, sr, smb_fqi_t *, fqi);

 109         return ((rc == 0) ? SDRC_SUCCESS : SDRC_ERROR);
 110 }
______unchanged_portion_omitted_
```

```
**********************************************************
   12798 Fri Mar 31 14:29:39 2017
new/usr/src/uts/common/fs/smbsrv/smb_directory.c
Build provider 3rd arg from smb_request_t
hacking...
**********************************************************
```

```
/*
 * 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 2010 Sun Microsystems, Inc.  All rights reserved.
 * Use is subject to license terms.
 *
 * Copyright 2013 Nexenta Systems, Inc.  All rights reserved.
 */

#include <smbsrv/smb_kproto.h>
#include <smbsrv/smbinfo.h>
#include <smbsrv/smb_fsops.h>

/*
 * The create directory message is sent to create a new directory.  The
 * appropriate Tid and additional pathname are passed.  The directory must
 * not exist for it to be created.
 *
 * Client Request                     Description
 * ================================== ================================
 * UCHAR WordCount;                   Count of parameter words = 0
 * USHORT ByteCount;                  Count of data bytes; min = 2
 * UCHAR BufferFormat;                0x04
 * STRING DirectoryName[];            Directory name
 *
 * Servers require clients to have at least create permission for the
 * subtree containing the directory in order to create a new directory.
 * The creator's access rights to the new directory are be determined by
 * local policy on the server.
 *
 * Server Response                    Description
 * ================================== ================================
 * UCHAR WordCount;                   Count of parameter words = 0
 * USHORT ByteCount;                  Count of data bytes = 0
 */
smb_sdrc_t
smb_pre_create_directory(smb_request_t *sr)
{
        int rc;

        rc = smbsr_decode_data(sr, "%S", sr,
            &sr->arg.dirop.fqi.fq_path.pn_path);
```



```
        DTRACE_SMB_1(op__CreateDirectory__start, smb_request_t *, sr); /* arg.di
        DTRACE_SMB_2(op__CreateDirectory__start, smb_request_t *, sr,
            struct dirop *, &sr->arg.dirop);

        return ((rc == 0) ? SDRC_SUCCESS : SDRC_ERROR);
}
```

*unchanged_portion_omitted_*

```
/*
 * The delete directory message is sent to delete an empty directory. The
 * appropriate Tid and additional pathname are passed. The directory must
 * be empty for it to be deleted.
 *
 * NT supports a hidden permission known as File Delete Child (FDC). If
 * the user has FullControl access to a directory, the user is permitted
 * to delete any object in the directory regardless of the permissions
 * on the object.
 *
 * Client Request                     Description
 * ================================== ================================
 * UCHAR WordCount;                   Count of parameter words = 0
 * USHORT ByteCount;                  Count of data bytes; min = 2
 * UCHAR BufferFormat;                0x04
 * STRING DirectoryName[];            Directory name
 *
 * The directory to be deleted cannot be the root of the share specified
 * by Tid.
 *
 * Server Response                    Description
 * ================================== ================================
 * UCHAR WordCount;                   Count of parameter words = 0
 * USHORT ByteCount;                  Count of data bytes = 0
 */
smb_sdrc_t
smb_pre_delete_directory(smb_request_t *sr)
{
        int rc;

        rc = smbsr_decode_data(sr, "%S", sr,
            &sr->arg.dirop.fqi.fq_path.pn_path);

        DTRACE_SMB_1(op__DeleteDirectory__start, smb_request_t *, sr); /* arg.di
        DTRACE_SMB_2(op__DeleteDirectory__start, smb_request_t *, sr,
            struct dirop *, &sr->arg.dirop);

        return ((rc == 0) ? SDRC_SUCCESS : SDRC_ERROR);
}
```

*unchanged_portion_omitted_*

```
/*
 * This SMB is used to verify that a path exists and is a directory.  No
 * error is returned if the given path exists and the client has read
 * access to it.  Client machines which maintain a concept of a "working
 * directory" will find this useful to verify the validity of a "change
 * working directory" command.  Note that the servers do NOT have a concept
 * of working directory for a particular client.  The client must always
 * supply full pathnames relative to the Tid in the SMB header.
 *
 * Client Request                     Description
 * ================================== ================================
 *
 * UCHAR WordCount;                   Count of parameter words = 0
 * USHORT ByteCount;                  Count of data bytes;    min = 2
 * UCHAR BufferFormat;                0x04
 * STRING DirectoryPath[];            Directory path
```

```
 344  *
 345  * Server Response                    Description
 346  * ================================= ================================
 347  *
 348  * UCHAR WordCount;                   Count of parameter words = 0
 349  * USHORT ByteCount;                  Count of data bytes = 0
 350  *
 351  * DOS clients, in particular, depend on ERRbadpath if the directory is
 352  * not found.
 353  */
 354 smb_sdrc_t
 355 smb_pre_check_directory(smb_request_t *sr)
 356 {
 357         int rc;

 359         rc = smbsr_decode_data(sr, "%S", sr,
 360             &sr->arg.dirop.fqi.fq_path.pn_path);

 362         DTRACE_SMB_1(op__CheckDirectory__start, smb_request_t *, sr); /* arg.dir
 364         DTRACE_SMB_2(op__CheckDirectory__start, smb_request_t *, sr,
 365             struct dirop *, &sr->arg.dirop);

 364         return ((rc == 0) ? SDRC_SUCCESS : SDRC_ERROR);
 365 }
```

*unchanged_portion_omitted_*

```
**********************************************************
   45220 Fri Mar 31 14:29:40 2017
new/usr/src/uts/common/fs/smbsrv/smb_dispatch.c
NEX-1643 dtrace provider for smbsrv
Also illumos 1841:
 DTrace smb provider was mis-implemented, doesn't exist.
Add back handlers for read/write raw, so that
 legacy dtrace consumers can find the probes.
Kill extra arg in smb_negotiate
Fix missing "done" probe with smb_notify
Add example consumer: smb-trace.d
fix soi_pid
**********************************************************
```

```
/*
 * 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) 2007, 2010, Oracle and/or its affiliates. All rights reserved.
 * Copyright 2016 Nexenta Systems, Inc.  All rights reserved.
 */

/*
 * SMB requests.
 *
 * Request
 *   Header
 *      Magic           0xFF 'S' 'M' 'B'
 *      smb_com         a byte, the "first" command
 *      Error           a 4-byte union, ignored in a request
 *      smb_flg         a one byte set of eight flags
 *      smb_flg2        a two byte set of 16 flags
 *      .               twelve reserved bytes, have a role
 *                      in connectionless transports (IPX, UDP?)
 *      smb_tid         a 16-bit tree ID, a mount point sorta,
 *                      0xFFFF is this command does not have
 *                      or require a tree context
 *      smb_pid         a 16-bit process ID
 *      smb_uid         a 16-bit user ID, specific to this "session"
 *                      and mapped to a system (bona-fide) UID
 *      smb_mid         a 16-bit multiplex ID, used to differentiate
 *                      multiple simultaneous requests from the same
 *                      process (pid) (ref RPC "xid")
 *
 *   Chained (AndX) commands (0 or more)
 *      smb_wct         a byte, number of 16-bit words containing
 *                      command parameters, min 2 for chained command
 *      andx_com        a byte, the "next" command, 0xFF for none
 *      .               an unused byte
```



```
 *      andx_off        a 16-bit offset, byte displacement from &Magic
 *                      to the smb_wct field of the "next" command,
 *                      ignore if andx_com is 0xFF, s/b 0 if no next
 *      smb_vwv[]       0 or more 16-bit (sorta) parameters for
 *                      "this" command (i.e. smb_com if this is the
 *                      first parameters, or the andx_com of the just
 *                      previous block.
 *      smb_bcc         a 16-bit count of smb_data[] bytes
 *      smb_data[]      0 or more bytes, format specific to commands
 *      padding[]       Optional padding
 *
 *   Last command
 *      smb_wct         a byte, number of 16-bit words containing
 *                      command parameters, min 0 for chained command
 *      smb_vwv[]       0 or more 16-bit (sorta) parameters for
 *                      "this" command (i.e. smb_com if this is the
 *                      first parameters, or the andx_com of the just
 *                      previous block.
 *      smb_bcc         a 16-bit count of smb_data[] bytes
 *      smb_data[]      0 or more bytes, format specific to commands
 *
 * Reply
 *   Header
 *      Magic           0xFF 'S' 'M' 'B'
 *      smb_com         a byte, the "first" command, corresponds
 *                      to request
 *      Error           a 4-byte union, coding depends on dialect in use
 *                      for "DOS" errors
 *                              a byte for error class
 *                              an unused byte
 *                              a 16-bit word for error code
 *                      for "NT" errors
 *                              a 32-bit error code which
 *                              is a packed class and specifier
 *                      for "OS/2" errors
 *                              I don't know
 *                      The error information is specific to the
 *                      last command in the reply chain.
 *      smb_flg         a one byte set of eight flags, 0x80 bit set
 *                      indicating this message is a reply
 *      smb_flg2        a two byte set of 16 flags
 *      .               twelve reserved bytes, have a role
 *                      in connectionless transports (IPX, UDP?)
 *      smb_tid         a 16-bit tree ID, a mount point sorta,
 *                      should be the same as the request
 *      smb_pid         a 16-bit process ID, MUST BE the same as request
 *      smb_uid         a 16-bit user ID, specific to this "session"
 *                      and mapped to a system (bona-fide) UID,
 *                      should be the same as request
 *      smb_mid         a 16-bit multiplex ID, used to differentiate
 *                      multiple simultaneous requests from the same
 *                      process (pid) (ref RPC "xid"), MUST BE the
 *                      same as request
 *      padding[]       Optional padding
 *
 *   Chained (AndX) commands (0 or more)
 *      smb_wct         a byte, number of 16-bit words containing
 *                      command parameters, min 2 for chained command,
 *      andx_com        a byte, the "next" command, 0xFF for none,
 *                      corresponds to request, if this is the chained
 *                      command that had an error set to 0xFF
 *      .               an unused byte
 *      andx_off        a 16-bit offset, byte displacement from &Magic
 *                      to the smb_wct field of the "next" command,
 *                      ignore if andx_com is 0xFF, s/b 0 if no next
 *      smb_vwv[]       0 or more 16-bit (sorta) parameters for
```

```
 120  *                      "this" command (i.e. smb_com if this is the
 121  *                      first parameters, or the andx_com of the just
 122  *                      previous block. Empty if an error.
 123  *      smb_bcc         a 16-bit count of smb_data[] bytes
 124  *      smb_data[]      0 or more bytes, format specific to commands
 125  *                      empty if an error.
 126  *
 127  *   Last command
 128  *      smb_wct         a byte, number of 16-bit words containing
 129  *                      command parameters, min 0 for chained command
 130  *      smb_vwv[]       0 or more 16-bit (sorta) parameters for
 131  *                      "this" command (i.e. smb_com if this is the
 132  *                      first parameters, or the andx_com of the just
 133  *                      previous block, empty if an error.
 134  *      smb_bcc         a 16-bit count of smb_data[] bytes
 135  *      smb_data[]      0 or more bytes, format specific to commands,
 136  *                      empty if an error.
 137  */

 139 #include <smbsrv/smb_kproto.h>
 140 #include <smbsrv/smb_kstat.h>
 141 #include <sys/sdt.h>
 142 #include <sys/spl.h>

 144 static int is_andx_com(unsigned char);
 145 static int smbsr_check_result(struct smb_request *, int, int);
 146 static void smb1_tq_work(void *);

 148 static const smb_disp_entry_t const
 149 smb_disp_table[SMB_COM_NUM] = {
 150         { "SmbCreateDirectory", SMB_SDT_OPS(create_directory),  /* 0x00 000 */
 151           0x00, PC_NETWORK_PROGRAM_1_0 },
 152         { "SmbDeleteDirectory", SMB_SDT_OPS(delete_directory),  /* 0x01 001 */
 153           0x01, PC_NETWORK_PROGRAM_1_0 },
 154         { "SmbOpen", SMB_SDT_OPS(open),                         /* 0x02 002 */
 155           0x02, PC_NETWORK_PROGRAM_1_0 },
 156         { "SmbCreate", SMB_SDT_OPS(create),                     /* 0x03 003 */
 157           0x03, PC_NETWORK_PROGRAM_1_0 },
 158         { "SmbClose", SMB_SDT_OPS(close),                       /* 0x04 004 */
 159           0x04, PC_NETWORK_PROGRAM_1_0 },
 160         { "SmbFlush", SMB_SDT_OPS(flush),                       /* 0x05 005 */
 161           0x05, PC_NETWORK_PROGRAM_1_0 },
 162         { "SmbDelete", SMB_SDT_OPS(delete),                     /* 0x06 006 */
 163           0x06, PC_NETWORK_PROGRAM_1_0 },
 164         { "SmbRename", SMB_SDT_OPS(rename),                     /* 0x07 007 */
 165           0x07, PC_NETWORK_PROGRAM_1_0 },
 166         { "SmbQueryInformation", SMB_SDT_OPS(query_information), /* 0x08 008 */
 167           0x08, PC_NETWORK_PROGRAM_1_0 },
 168         { "SmbSetInformation", SMB_SDT_OPS(set_information),    /* 0x09 009 */
 169           0x09, PC_NETWORK_PROGRAM_1_0 },
 170         { "SmbRead", SMB_SDT_OPS(read),                         /* 0x0A 010 */
 171           0x0A, PC_NETWORK_PROGRAM_1_0, SDDF_READOP },
 172         { "SmbWrite", SMB_SDT_OPS(write),                       /* 0x0B 011 */
 173           0x0B, PC_NETWORK_PROGRAM_1_0, SDDF_WRITEOP },
 174         { "SmbLockByteRange", SMB_SDT_OPS(lock_byte_range),     /* 0x0C 012 */
 175           0x0C, PC_NETWORK_PROGRAM_1_0 },
 176         { "SmbUnlockByteRange", SMB_SDT_OPS(unlock_byte_range), /* 0x0D 013 */
 177           0x0D, PC_NETWORK_PROGRAM_1_0 },
 178         { "SmbCreateTemporary", SMB_SDT_OPS(create_temporary),  /* 0x0E 014 */
 179           0x0E, PC_NETWORK_PROGRAM_1_0 },
 180         { "SmbCreateNew", SMB_SDT_OPS(create_new),              /* 0x0F 015 */
 181           0x0F, PC_NETWORK_PROGRAM_1_0 },
 182         { "SmbCheckDirectory", SMB_SDT_OPS(check_directory),    /* 0x10 016 */
 183           0x10, PC_NETWORK_PROGRAM_1_0 },
 184         { "SmbProcessExit", SMB_SDT_OPS(process_exit),          /* 0x11 017 */
 185           0x11, PC_NETWORK_PROGRAM_1_0,
```



```
 186           SDDF_SUPPRESS_TID | SDDF_SUPPRESS_UID },
 187         { "SmbSeek", SMB_SDT_OPS(seek),                         /* 0x12 018 */
 188           0x12, PC_NETWORK_PROGRAM_1_0 },
 189         { "SmbLockAndRead", SMB_SDT_OPS(lock_and_read),         /* 0x13 019 */
 190           0x13, LANMAN1_0, SDDF_READOP},
 191         { "SmbWriteAndUnlock", SMB_SDT_OPS(write_and_unlock),   /* 0x14 020 */
 192           0x14, LANMAN1_0, SDDF_WRITEOP },
 193         { "Invalid", SMB_SDT_OPS(invalid), 0x15, 0 },           /* 0x15 021 */
 194         { "Invalid", SMB_SDT_OPS(invalid), 0x16, 0 },           /* 0x16 022 */
 195         { "Invalid", SMB_SDT_OPS(invalid), 0x17, 0 },           /* 0x17 023 */
 196         { "Invalid", SMB_SDT_OPS(invalid), 0x18, 0 },           /* 0x18 024 */
 197         { "Invalid", SMB_SDT_OPS(invalid), 0x19, 0 },           /* 0x19 025 */
 198         { "SmbReadRaw", SMB_SDT_OPS(read_raw),                  /* 0x1A 026 */
 199           0x1A, LANMAN1_0 },
 198         { "SmbReadRaw", SMB_SDT_OPS(invalid), 0x1A, 0 },        /* 0x1A 026 */
 200         { "Invalid", SMB_SDT_OPS(invalid), 0x1B, 0 },           /* 0x1B 027 */
 201         { "Invalid", SMB_SDT_OPS(invalid), 0x1C, 0 },           /* 0x1C 028 */
 202         { "SmbWriteRaw", SMB_SDT_OPS(write_raw),                /* 0x1D 029 */
 203           0x1D, LANMAN1_0 },
 201         { "SmbWriteRaw", SMB_SDT_OPS(invalid), 0x1D, 0 },       /* 0x1D 029 */
 204         { "Invalid", SMB_SDT_OPS(invalid), 0x1E, 0 },           /* 0x1E 030 */
 205         { "Invalid", SMB_SDT_OPS(invalid), 0x1F, 0 },           /* 0x1F 031 */
 206         { "Invalid", SMB_SDT_OPS(invalid), 0x20, 0 },           /* 0x20 032 */
 207         { "Invalid", SMB_SDT_OPS(invalid), 0x21, 0 },           /* 0x21 033 */
 208         { "SmbSetInformation2", SMB_SDT_OPS(set_information2),  /* 0x22 034 */
 209           0x22, LANMAN1_0 },
 210         { "SmbQueryInformation2",
 211           SMB_SDT_OPS(query_information2),                      /* 0x23 035 */
 212           0x23, LANMAN1_0 },
 213         { "SmbLockingX", SMB_SDT_OPS(locking_andx),             /* 0x24 036 */
 214           0x24, LANMAN1_0 },
 215         { "SmbTransaction", SMB_SDT_OPS(transaction),           /* 0x25 037 */
 216           0x25, LANMAN1_0 },
 217         { "SmbTransactionSecondary",
 218           SMB_SDT_OPS(transaction_secondary),                   /* 0x26 038 */
 219           0x26, LANMAN1_0 },
 220         { "SmbIoctl", SMB_SDT_OPS(ioctl),                       /* 0x27 039 */
 221           0x27, LANMAN1_0 },
 222         { "Invalid", SMB_SDT_OPS(invalid), 0x28, 0 },   /* 0x28 040 */
 223         { "Invalid", SMB_SDT_OPS(invalid), 0x29, 0 },   /* 0x29 041 */
 224         { "Invalid", SMB_SDT_OPS(invalid), 0x2A, 0 },   /* 0x2A 042 */
 225         { "SmbEcho", SMB_SDT_OPS(echo),                         /* 0x2B 043 */
 226           0x2B, LANMAN1_0, SDDF_SUPPRESS_TID | SDDF_SUPPRESS_UID },
 227         { "SmbWriteAndClose", SMB_SDT_OPS(write_and_close),     /* 0x2C 044 */
 228           0x2C, LANMAN1_0, SDDF_WRITEOP },
 229         { "SmbOpenX", SMB_SDT_OPS(open_andx),                   /* 0x2D 045 */
 230           0x2D, LANMAN1_0 },
 231         { "SmbReadX", SMB_SDT_OPS(read_andx),                   /* 0x2E 046 */
 232           0x2E, LANMAN1_0, SDDF_READOP },
 233         { "SmbWriteX", SMB_SDT_OPS(write_andx),                 /* 0x2F 047 */
 234           0x2F, LANMAN1_0, SDDF_WRITEOP },
 235         { "Invalid", SMB_SDT_OPS(invalid), 0x30, 0 },   /* 0x30 048 */
 236         { "SmbCloseAndTreeDisconnect",
 237           SMB_SDT_OPS(close_and_tree_disconnect),               /* 0x31 049 */
 238           0x31, LANMAN1_0 },
 239         { "SmbTransaction2", SMB_SDT_OPS(transaction2),         /* 0x32 050 */
 240           0x32, LM1_2X002 },
 241         { "SmbTransaction2Secondary",
 242           SMB_SDT_OPS(transaction2_secondary),                  /* 0x33 051 */
 243           0x33, LM1_2X002 },
 244         { "SmbFindClose2", SMB_SDT_OPS(find_close2),            /* 0x34 052 */
 245           0x34, LM1_2X002 },
 246         { "Invalid", SMB_SDT_OPS(invalid), 0x35, 0 },   /* 0x35 053 */
 247         { "Invalid", SMB_SDT_OPS(invalid), 0x36, 0 },   /* 0x36 054 */
 248         { "Invalid", SMB_SDT_OPS(invalid), 0x37, 0 },   /* 0x37 055 */
 249         { "Invalid", SMB_SDT_OPS(invalid), 0x38, 0 },   /* 0x38 056 */
```

```
250         { "Invalid", SMB_SDT_OPS(invalid), 0x39, 0 },   /* 0x39 057 */
251         { "Invalid", SMB_SDT_OPS(invalid), 0x3A, 0 },   /* 0x3A 058 */
252         { "Invalid", SMB_SDT_OPS(invalid), 0x3B, 0 },   /* 0x3B 059 */
253         { "Invalid", SMB_SDT_OPS(invalid), 0x3C, 0 },   /* 0x3C 060 */
254         { "Invalid", SMB_SDT_OPS(invalid), 0x3D, 0 },   /* 0x3D 061 */
255         { "Invalid", SMB_SDT_OPS(invalid), 0x3E, 0 },   /* 0x3E 062 */
256         { "Invalid", SMB_SDT_OPS(invalid), 0x3F, 0 },   /* 0x3F 063 */
257         { "Invalid", SMB_SDT_OPS(invalid), 0x40, 0 },   /* 0x40 064 */
258         { "Invalid", SMB_SDT_OPS(invalid), 0x47, 0 },   /* 0x47 065 */
259         { "Invalid", SMB_SDT_OPS(invalid), 0x48, 0 },   /* 0x48 066 */
260         { "Invalid", SMB_SDT_OPS(invalid), 0x49, 0 },   /* 0x49 067 */
261         { "Invalid", SMB_SDT_OPS(invalid), 0x44, 0 },   /* 0x44 068 */
262         { "Invalid", SMB_SDT_OPS(invalid), 0x45, 0 },   /* 0x45 069 */
263         { "Invalid", SMB_SDT_OPS(invalid), 0x46, 0 },   /* 0x46 070 */
264         { "Invalid", SMB_SDT_OPS(invalid), 0x47, 0 },   /* 0x47 071 */
265         { "Invalid", SMB_SDT_OPS(invalid), 0x48, 0 },   /* 0x48 072 */
266         { "Invalid", SMB_SDT_OPS(invalid), 0x49, 0 },   /* 0x49 073 */
267         { "Invalid", SMB_SDT_OPS(invalid), 0x4A, 0 },   /* 0x4A 074 */
268         { "Invalid", SMB_SDT_OPS(invalid), 0x4B, 0 },   /* 0x4B 075 */
269         { "Invalid", SMB_SDT_OPS(invalid), 0x4C, 0 },   /* 0x4C 076 */
270         { "Invalid", SMB_SDT_OPS(invalid), 0x4D, 0 },   /* 0x4D 077 */
271         { "Invalid", SMB_SDT_OPS(invalid), 0x4E, 0 },   /* 0x4E 078 */
272         { "Invalid", SMB_SDT_OPS(invalid), 0x4F, 0 },   /* 0x4F 079 */
273         { "Invalid", SMB_SDT_OPS(invalid), 0x50, 0 },   /* 0x50 080 */
274         { "Invalid", SMB_SDT_OPS(invalid), 0x51, 0 },   /* 0x51 081 */
275         { "Invalid", SMB_SDT_OPS(invalid), 0x52, 0 },   /* 0x52 082 */
276         { "Invalid", SMB_SDT_OPS(invalid), 0x53, 0 },   /* 0x53 083 */
277         { "Invalid", SMB_SDT_OPS(invalid), 0x54, 0 },   /* 0x54 084 */
278         { "Invalid", SMB_SDT_OPS(invalid), 0x55, 0 },   /* 0x55 085 */
279         { "Invalid", SMB_SDT_OPS(invalid), 0x56, 0 },   /* 0x56 086 */
280         { "Invalid", SMB_SDT_OPS(invalid), 0x57, 0 },   /* 0x57 087 */
281         { "Invalid", SMB_SDT_OPS(invalid), 0x58, 0 },   /* 0x58 088 */
282         { "Invalid", SMB_SDT_OPS(invalid), 0x59, 0 },   /* 0x59 089 */
283         { "Invalid", SMB_SDT_OPS(invalid), 0x5A, 0 },   /* 0x5A 090 */
284         { "Invalid", SMB_SDT_OPS(invalid), 0x5B, 0 },   /* 0x5B 091 */
285         { "Invalid", SMB_SDT_OPS(invalid), 0x5C, 0 },   /* 0x5C 092 */
286         { "Invalid", SMB_SDT_OPS(invalid), 0x5D, 0 },   /* 0x5D 093 */
287         { "Invalid", SMB_SDT_OPS(invalid), 0x5E, 0 },   /* 0x5E 094 */
288         { "Invalid", SMB_SDT_OPS(invalid), 0x5F, 0 },   /* 0x5F 095 */
289         { "Invalid", SMB_SDT_OPS(invalid), 0x60, 0 },   /* 0x60 096 */
290         { "Invalid", SMB_SDT_OPS(invalid), 0x61, 0 },   /* 0x61 097 */
291         { "Invalid", SMB_SDT_OPS(invalid), 0x62, 0 },   /* 0x62 098 */
292         { "Invalid", SMB_SDT_OPS(invalid), 0x63, 0 },   /* 0x63 099 */
293         { "Invalid", SMB_SDT_OPS(invalid), 0x64, 0 },   /* 0x64 100 */
294         { "Invalid", SMB_SDT_OPS(invalid), 0x65, 0 },   /* 0x65 101 */
295         { "Invalid", SMB_SDT_OPS(invalid), 0x66, 0 },   /* 0x66 102 */
296         { "Invalid", SMB_SDT_OPS(invalid), 0x67, 0 },   /* 0x67 103 */
297         { "Invalid", SMB_SDT_OPS(invalid), 0x68, 0 },   /* 0x68 104 */
298         { "Invalid", SMB_SDT_OPS(invalid), 0x69, 0 },   /* 0x69 105 */
299         { "Invalid", SMB_SDT_OPS(invalid), 0x6A, 0 },   /* 0x6A 106 */
300         { "Invalid", SMB_SDT_OPS(invalid), 0x6B, 0 },   /* 0x6B 107 */
301         { "Invalid", SMB_SDT_OPS(invalid), 0x6C, 0 },   /* 0x6C 108 */
302         { "Invalid", SMB_SDT_OPS(invalid), 0x6D, 0 },   /* 0x6D 109 */
303         { "Invalid", SMB_SDT_OPS(invalid), 0x6E, 0 },   /* 0x6E 110 */
304         { "Invalid", SMB_SDT_OPS(invalid), 0x6F, 0 },   /* 0x6F 111 */
305         { "SmbTreeConnect", SMB_SDT_OPS(tree_connect),          /* 0x70 112 */
306             0x70, PC_NETWORK_PROGRAM_1_0, SDDF_SUPPRESS_TID },
307         { "SmbTreeDisconnect", SMB_SDT_OPS(tree_disconnect),    /* 0x71 113 */
308             0x71, PC_NETWORK_PROGRAM_1_0,
309             SDDF_SUPPRESS_TID | SDDF_SUPPRESS_UID },
310         /*
311          * NB: Negotiate gets special handling via
312          * smb_initial_request_handler.  After that,
313          * another Negotiate is an invalid request.
314          */
315         { "SmbNegotiate", SMB_SDT_OPS(invalid),                 /* 0x72 114 */
```



```
316             0x72, PC_NETWORK_PROGRAM_1_0,
317             SDDF_SUPPRESS_TID | SDDF_SUPPRESS_UID },
318         { "SmbSessionSetupX", SMB_SDT_OPS(session_setup_andx),  /* 0x73 115 */
319             0x73, LANMAN1_0, SDDF_SUPPRESS_TID | SDDF_SUPPRESS_UID },
320         { "SmbLogoffX", SMB_SDT_OPS(logoff_andx),               /* 0x74 116 */
321             0x74, LM1_2X002, SDDF_SUPPRESS_TID },
322         { "SmbTreeConnectX", SMB_SDT_OPS(tree_connect_andx),    /* 0x75 117 */
323             0x75, LANMAN1_0, SDDF_SUPPRESS_TID },
324         { "Invalid", SMB_SDT_OPS(invalid), 0x76, 0, 0 },        /* 0x76 118 */
325         { "Invalid", SMB_SDT_OPS(invalid), 0x77, 0, 0 },        /* 0x77 119 */
326         { "Invalid", SMB_SDT_OPS(invalid), 0x78, 0, 0 },        /* 0x78 120 */
327         { "Invalid", SMB_SDT_OPS(invalid), 0x79, 0, 0 },        /* 0x79 121 */
328         { "Invalid", SMB_SDT_OPS(invalid), 0x7A, 0, 0 },        /* 0x7A 122 */
329         { "Invalid", SMB_SDT_OPS(invalid), 0x7B, 0, 0 },        /* 0x7B 123 */
330         { "Invalid", SMB_SDT_OPS(invalid), 0x7C, 0, 0 },        /* 0x7C 124 */
331         { "Invalid", SMB_SDT_OPS(invalid), 0x7D, 0, 0 },        /* 0x7D 125 */
332         { "Invalid", SMB_SDT_OPS(invalid), 0x7E, 0, 0 },        /* 0x7E 126 */
333         { "Invalid", SMB_SDT_OPS(invalid), 0x7F, 0, 0 },        /* 0x7F 127 */
334         { "SmbQueryInformationDisk",
335             SMB_SDT_OPS(query_information_disk),                /* 0x80 128 */
336             0x80, PC_NETWORK_PROGRAM_1_0, 0 },
337         { "SmbSearch", SMB_SDT_OPS(search),                     /* 0x81 129 */
338             0x81, PC_NETWORK_PROGRAM_1_0, 0 },
339         { "SmbFind", SMB_SDT_OPS(find),                         /* 0x82 130 */
340             0x82, LANMAN1_0, 0 },
341         { "SmbFindUnique", SMB_SDT_OPS(find_unique),            /* 0x83 131 */
342             0x83, LANMAN1_0, 0 },
343         { "SmbFindClose", SMB_SDT_OPS(find_close),              /* 0x84 132 */
344             0x84, LANMAN1_0, 0 },
345         { "Invalid", SMB_SDT_OPS(invalid), 0x85, 0, 0 },        /* 0x85 133 */
346         { "Invalid", SMB_SDT_OPS(invalid), 0x86, 0, 0 },        /* 0x86 134 */
347         { "Invalid", SMB_SDT_OPS(invalid), 0x87, 0, 0 },        /* 0x87 135 */
348         { "Invalid", SMB_SDT_OPS(invalid), 0x88, 0, 0 },        /* 0x88 136 */
349         { "Invalid", SMB_SDT_OPS(invalid), 0x89, 0, 0 },        /* 0x89 137 */
350         { "Invalid", SMB_SDT_OPS(invalid), 0x8A, 0, 0 },        /* 0x8A 138 */
351         { "Invalid", SMB_SDT_OPS(invalid), 0x8B, 0, 0 },        /* 0x8B 139 */
352         { "Invalid", SMB_SDT_OPS(invalid), 0x8C, 0, 0 },        /* 0x8C 140 */
353         { "Invalid", SMB_SDT_OPS(invalid), 0x8D, 0, 0 },        /* 0x8D 141 */
354         { "Invalid", SMB_SDT_OPS(invalid), 0x8E, 0, 0 },        /* 0x8E 142 */
355         { "Invalid", SMB_SDT_OPS(invalid), 0x8F, 0, 0 },        /* 0x8F 143 */
356         { "Invalid", SMB_SDT_OPS(invalid), 0x90, 0, 0 },        /* 0x90 144 */
357         { "Invalid", SMB_SDT_OPS(invalid), 0x91, 0, 0 },        /* 0x91 145 */
358         { "Invalid", SMB_SDT_OPS(invalid), 0x92, 0, 0 },        /* 0x92 146 */
359         { "Invalid", SMB_SDT_OPS(invalid), 0x93, 0, 0 },        /* 0x93 147 */
360         { "Invalid", SMB_SDT_OPS(invalid), 0x94, 0, 0 },        /* 0x94 148 */
361         { "Invalid", SMB_SDT_OPS(invalid), 0x95, 0, 0 },        /* 0x95 149 */
362         { "Invalid", SMB_SDT_OPS(invalid), 0x96, 0, 0 },        /* 0x96 150 */
363         { "Invalid", SMB_SDT_OPS(invalid), 0x97, 0, 0 },        /* 0x97 151 */
364         { "Invalid", SMB_SDT_OPS(invalid), 0x98, 0, 0 },        /* 0x98 152 */
365         { "Invalid", SMB_SDT_OPS(invalid), 0x99, 0, 0 },        /* 0x99 153 */
366         { "Invalid", SMB_SDT_OPS(invalid), 0x9A, 0, 0 },        /* 0x9A 154 */
367         { "Invalid", SMB_SDT_OPS(invalid), 0x9B, 0, 0 },        /* 0x9B 155 */
368         { "Invalid", SMB_SDT_OPS(invalid), 0x9C, 0, 0 },        /* 0x9C 156 */
369         { "Invalid", SMB_SDT_OPS(invalid), 0x9D, 0, 0 },        /* 0x9D 157 */
370         { "Invalid", SMB_SDT_OPS(invalid), 0x9E, 0, 0 },        /* 0x9E 158 */
371         { "Invalid", SMB_SDT_OPS(invalid), 0x9F, 0, 0 },        /* 0x9F 159 */
372         { "SmbNtTransact", SMB_SDT_OPS(nt_transact),    /* 0xA0 160 */
373             0xA0, NT_LM_0_12, 0 },
374         { "SmbNtTransactSecondary",
375             SMB_SDT_OPS(nt_transact_secondary),                 /* 0xA1 161 */
376             0xA1, NT_LM_0_12, 0 },
377         { "SmbNtCreateX", SMB_SDT_OPS(nt_create_andx),          /* 0xA2 162 */
378             0xA2, NT_LM_0_12, 0 },
379         { "Invalid", SMB_SDT_OPS(invalid), 0xA3, 0, 0 },        /* 0xA3 163 */
380         { "SmbNtCancel", SMB_SDT_OPS(nt_cancel),                /* 0xA4 164 */
381             0xA4, NT_LM_0_12, 0 },
```

```
382         { "SmbNtRename", SMB_SDT_OPS(nt_rename),                /* 0xA5 165 */
383             0xA5, NT_LM_0_12, 0 },
384         { "Invalid", SMB_SDT_OPS(invalid), 0xA6, 0, 0 },        /* 0xA6 166 */
385         { "Invalid", SMB_SDT_OPS(invalid), 0xA7, 0, 0 },        /* 0xA7 167 */
386         { "Invalid", SMB_SDT_OPS(invalid), 0xA8, 0, 0 },        /* 0xA8 168 */
387         { "Invalid", SMB_SDT_OPS(invalid), 0xA9, 0, 0 },        /* 0xA9 169 */
388         { "Invalid", SMB_SDT_OPS(invalid), 0xAA, 0, 0 },        /* 0xAA 170 */
389         { "Invalid", SMB_SDT_OPS(invalid), 0xAB, 0, 0 },        /* 0xAB 171 */
390         { "Invalid", SMB_SDT_OPS(invalid), 0xAC, 0, 0 },        /* 0xAC 172 */
391         { "Invalid", SMB_SDT_OPS(invalid), 0xAD, 0, 0 },        /* 0xAD 173 */
392         { "Invalid", SMB_SDT_OPS(invalid), 0xAE, 0, 0 },        /* 0xAE 174 */
393         { "Invalid", SMB_SDT_OPS(invalid), 0xAF, 0, 0 },        /* 0xAF 175 */
394         { "Invalid", SMB_SDT_OPS(invalid), 0xB0, 0, 0 },        /* 0xB0 176 */
395         { "Invalid", SMB_SDT_OPS(invalid), 0xB1, 0, 0 },        /* 0xB1 177 */
396         { "Invalid", SMB_SDT_OPS(invalid), 0xB2, 0, 0 },        /* 0xB2 178 */
397         { "Invalid", SMB_SDT_OPS(invalid), 0xB3, 0, 0 },        /* 0xB3 179 */
398         { "Invalid", SMB_SDT_OPS(invalid), 0xB4, 0, 0 },        /* 0xB4 180 */
399         { "Invalid", SMB_SDT_OPS(invalid), 0xB5, 0, 0 },        /* 0xB5 181 */
400         { "Invalid", SMB_SDT_OPS(invalid), 0xB6, 0, 0 },        /* 0xB6 182 */
401         { "Invalid", SMB_SDT_OPS(invalid), 0xB7, 0, 0 },        /* 0xB7 183 */
402         { "Invalid", SMB_SDT_OPS(invalid), 0xB8, 0, 0 },        /* 0xB8 184 */
403         { "Invalid", SMB_SDT_OPS(invalid), 0xB9, 0, 0 },        /* 0xB9 185 */
404         { "Invalid", SMB_SDT_OPS(invalid), 0xBA, 0, 0 },        /* 0xBA 186 */
405         { "Invalid", SMB_SDT_OPS(invalid), 0xBB, 0, 0 },        /* 0xBB 187 */
406         { "Invalid", SMB_SDT_OPS(invalid), 0xBC, 0, 0 },        /* 0xBC 188 */
407         { "Invalid", SMB_SDT_OPS(invalid), 0xBD, 0, 0 },        /* 0xBD 189 */
408         { "Invalid", SMB_SDT_OPS(invalid), 0xBE, 0, 0 },        /* 0xBE 190 */
409         { "Invalid", SMB_SDT_OPS(invalid), 0xBF, 0, 0 },        /* 0xBF 191 */
410         { "SmbOpenPrintFile", SMB_SDT_OPS(open_print_file),     /* 0xC0 192 */
411             0xC0, PC_NETWORK_PROGRAM_1_0, 0 },
412         { "SmbWritePrintFile", SMB_SDT_OPS(write_print_file),   /* 0xC1 193 */
413             0xC1, PC_NETWORK_PROGRAM_1_0, SDDF_WRITEOP },
414         { "SmbClosePrintFile", SMB_SDT_OPS(close_print_file),   /* 0xC2 194 */
415             0xC2, PC_NETWORK_PROGRAM_1_0, 0 },
416         { "SmbGetPrintQueue", SMB_SDT_OPS(get_print_queue),     /* 0xC3 195 */
417             0xC3, PC_NETWORK_PROGRAM_1_0, 0 },
418         { "Invalid", SMB_SDT_OPS(invalid), 0xC4, 0, 0 },        /* 0xC4 196 */
419         { "Invalid", SMB_SDT_OPS(invalid), 0xC5, 0, 0 },        /* 0xC5 197 */
420         { "Invalid", SMB_SDT_OPS(invalid), 0xC6, 0, 0 },        /* 0xC6 198 */
421         { "Invalid", SMB_SDT_OPS(invalid), 0xC7, 0, 0 },        /* 0xC7 199 */
422         { "Invalid", SMB_SDT_OPS(invalid), 0xC8, 0, 0 },        /* 0xC8 200 */
423         { "Invalid", SMB_SDT_OPS(invalid), 0xC9, 0, 0 },        /* 0xC9 201 */
424         { "Invalid", SMB_SDT_OPS(invalid), 0xCA, 0, 0 },        /* 0xCA 202 */
425         { "Invalid", SMB_SDT_OPS(invalid), 0xCB, 0, 0 },        /* 0xCB 203 */
426         { "Invalid", SMB_SDT_OPS(invalid), 0xCC, 0, 0 },        /* 0xCC 204 */
427         { "Invalid", SMB_SDT_OPS(invalid), 0xCD, 0, 0 },        /* 0xCD 205 */
428         { "Invalid", SMB_SDT_OPS(invalid), 0xCE, 0, 0 },        /* 0xCE 206 */
429         { "Invalid", SMB_SDT_OPS(invalid), 0xCF, 0, 0 },        /* 0xCF 207 */
430         { "Invalid", SMB_SDT_OPS(invalid), 0xD0, 0, 0 },        /* 0xD0 208 */
431         { "Invalid", SMB_SDT_OPS(invalid), 0xD1, 0, 0 },        /* 0xD1 209 */
432         { "Invalid", SMB_SDT_OPS(invalid), 0xD2, 0, 0 },        /* 0xD2 210 */
433         { "Invalid", SMB_SDT_OPS(invalid), 0xD3, 0, 0 },        /* 0xD3 211 */
434         { "Invalid", SMB_SDT_OPS(invalid), 0xD4, 0, 0 },        /* 0xD4 212 */
435         { "Invalid", SMB_SDT_OPS(invalid), 0xD5, 0, 0 },        /* 0xD5 213 */
436         { "Invalid", SMB_SDT_OPS(invalid), 0xD6, 0, 0 },        /* 0xD6 214 */
437         { "Invalid", SMB_SDT_OPS(invalid), 0xD7, 0, 0 },        /* 0xD7 215 */
438         { "Invalid", SMB_SDT_OPS(invalid), 0xD8, 0, 0 },        /* 0xD8 216 */
439         { "Invalid", SMB_SDT_OPS(invalid), 0xD9, 0, 0 },        /* 0xD9 217 */
440         { "Invalid", SMB_SDT_OPS(invalid), 0xDA, 0, 0 },        /* 0xDA 218 */
441         { "Invalid", SMB_SDT_OPS(invalid), 0xDB, 0, 0 },        /* 0xDB 219 */
442         { "Invalid", SMB_SDT_OPS(invalid), 0xDC, 0, 0 },        /* 0xDC 220 */
443         { "Invalid", SMB_SDT_OPS(invalid), 0xDD, 0, 0 },        /* 0xDD 221 */
444         { "Invalid", SMB_SDT_OPS(invalid), 0xDE, 0, 0 },        /* 0xDE 222 */
445         { "Invalid", SMB_SDT_OPS(invalid), 0xDF, 0, 0 },        /* 0xDF 223 */
446         { "Invalid", SMB_SDT_OPS(invalid), 0xE0, 0, 0 },        /* 0xE0 224 */
447         { "Invalid", SMB_SDT_OPS(invalid), 0xE1, 0, 0 },        /* 0xE1 225 */
```



```
448         { "Invalid", SMB_SDT_OPS(invalid), 0xE2, 0, 0 },        /* 0xE2 226 */
449         { "Invalid", SMB_SDT_OPS(invalid), 0xE3, 0, 0 },        /* 0xE3 227 */
450         { "Invalid", SMB_SDT_OPS(invalid), 0xE4, 0, 0 },        /* 0xE4 228 */
451         { "Invalid", SMB_SDT_OPS(invalid), 0xE5, 0, 0 },        /* 0xE5 229 */
452         { "Invalid", SMB_SDT_OPS(invalid), 0xE6, 0, 0 },        /* 0xE6 230 */
453         { "Invalid", SMB_SDT_OPS(invalid), 0xE7, 0, 0 },        /* 0xE7 231 */
454         { "Invalid", SMB_SDT_OPS(invalid), 0xE8, 0, 0 },        /* 0xE8 232 */
455         { "Invalid", SMB_SDT_OPS(invalid), 0xE9, 0, 0 },        /* 0xE9 233 */
456         { "Invalid", SMB_SDT_OPS(invalid), 0xEA, 0, 0 },        /* 0xEA 234 */
457         { "Invalid", SMB_SDT_OPS(invalid), 0xEB, 0, 0 },        /* 0xEB 235 */
458         { "Invalid", SMB_SDT_OPS(invalid), 0xEC, 0, 0 },        /* 0xEC 236 */
459         { "Invalid", SMB_SDT_OPS(invalid), 0xED, 0, 0 },        /* 0xED 237 */
460         { "Invalid", SMB_SDT_OPS(invalid), 0xEE, 0, 0 },        /* 0xEE 238 */
461         { "Invalid", SMB_SDT_OPS(invalid), 0xEF, 0, 0 },        /* 0xEF 239 */
462         { "Invalid", SMB_SDT_OPS(invalid), 0xF0, 0, 0 },        /* 0xF0 240 */
463         { "Invalid", SMB_SDT_OPS(invalid), 0xF1, 0, 0 },        /* 0xF1 241 */
464         { "Invalid", SMB_SDT_OPS(invalid), 0xF2, 0, 0 },        /* 0xF2 242 */
465         { "Invalid", SMB_SDT_OPS(invalid), 0xF3, 0, 0 },        /* 0xF3 243 */
466         { "Invalid", SMB_SDT_OPS(invalid), 0xF4, 0, 0 },        /* 0xF4 244 */
467         { "Invalid", SMB_SDT_OPS(invalid), 0xF5, 0, 0 },        /* 0xF5 245 */
468         { "Invalid", SMB_SDT_OPS(invalid), 0xF6, 0, 0 },        /* 0xF6 246 */
469         { "Invalid", SMB_SDT_OPS(invalid), 0xF7, 0, 0 },        /* 0xF7 247 */
470         { "Invalid", SMB_SDT_OPS(invalid), 0xF8, 0, 0 },        /* 0xF8 248 */
471         { "Invalid", SMB_SDT_OPS(invalid), 0xF9, 0, 0 },        /* 0xF9 249 */
472         { "Invalid", SMB_SDT_OPS(invalid), 0xFA, 0, 0 },        /* 0xFA 250 */
473         { "Invalid", SMB_SDT_OPS(invalid), 0xFB, 0, 0 },        /* 0xFB 251 */
474         { "Invalid", SMB_SDT_OPS(invalid), 0xFC, 0, 0 },        /* 0xFC 252 */
475         { "Invalid", SMB_SDT_OPS(invalid), 0xFD, 0, 0 },        /* 0xFD 253 */
476         { "Invalid", SMB_SDT_OPS(invalid), 0xFE, 0, 0 },        /* 0xFE 254 */
477         { "Invalid", SMB_SDT_OPS(invalid), 0xFF, 0, 0 } /* 0xFF 255 */
478 };
_____unchanged_portion_omitted_

633 /*
634  * smb1sr_work
635  *
636  * In most cases, this should free the request before return.
637  * Exceptions are when a dispatch function returns SDRC_SR_KEPT.
638  */
639 void
640 smb1sr_work(struct smb_request *sr)
641 {
642         smb_sdrc_t              sdrc;
643         const smb_disp_entry_t  *sdd;
644         smb_disp_stats_t        *sds;
645         boolean_t               disconnect = B_FALSE;
646         smb_session_t           *session;
647         smb_server_t            *server;
648         int32_t                 txbase;
649         uint32_t                rxbytes;
650         uint32_t                byte_count;
651         uint32_t                max_bytes;
652         uint16_t                pid_hi, pid_lo;

654         session = sr->session;
655         server = session->s_server;

657         ASSERT(sr->tid_tree == 0);
658         ASSERT(sr->uid_user == 0);
659         ASSERT(sr->fid_ofile == 0);
660         sr->smb_fid = (uint16_t)-1;

662         /* temporary until we identify a user */
663         sr->user_cr = zone_kcred();
664         sr->orig_request_hdr = sr->command.chain_offset;
```

```
666         /*
667          * Decode the SMB header.
668          */
669         if (smb_mbc_decodef(&sr->command, SMB_HEADER_ED_FMT,
670             &sr->smb_com,
671             &sr->smb_rcls,
672             &sr->smb_reh,
673             &sr->smb_err,
674             &sr->smb_flg,
675             &sr->smb_flg2,
676             &pid_hi,
677             sr->smb_sig,
678             &sr->smb_tid,
679             &pid_lo,
680             &sr->smb_uid,
681             &sr->smb_mid) != 0) {
682                 disconnect = B_TRUE;
683                 goto drop_connection;
684         }
685         sr->smb_pid = (pid_hi << 16) | pid_lo;

687         /*
688          * The reply "header" is filled in now even though it will,
689          * most likely, be rewritten under reply_ready below.  We
690          * could reserve the space but this is convenient in case
691          * the dialect dispatcher has to send a special reply (like
692          * TRANSACT).
693          *
694          * Ensure that the 32-bit error code flag is turned off.
695          * Clients seem to set it in transact requests and they may
696          * get confused if we return success or a 16-bit SMB code.
697          */
698         sr->smb_rcls = 0;
699         sr->smb_reh = 0;
700         sr->smb_err = 0;
701         sr->smb_flg2 &= ~SMB_FLAGS2_NT_STATUS;

703         (void) smb_mbc_encodef(&sr->reply, SMB_HEADER_ED_FMT,
704             sr->smb_com,
705             sr->smb_rcls,
706             sr->smb_reh,
707             sr->smb_err,
708             sr->smb_flg,
709             sr->smb_flg2,
710             pid_hi,
711             sr->smb_sig,
712             sr->smb_tid,
713             pid_lo,
714             sr->smb_uid,
715             sr->smb_mid);
716         sr->first_smb_com = sr->smb_com;

718         if ((session->signing.flags & SMB_SIGNING_CHECK) != 0) {
719                 if ((sr->smb_flg2 & SMB_FLAGS2_SMB_SECURITY_SIGNATURE) == 0 ||
720                     smb_sign_check_request(sr) != 0) {
721                         smbsr_error(sr, NT_STATUS_ACCESS_DENIED,
722                             ERRDOS, ERROR_ACCESS_DENIED);
723                         disconnect = B_TRUE;
724                         goto report_error;
725                 }
726         }

728 andx_more:
729         sdd = &smb_disp_table[sr->smb_com];
730         ASSERT(sdd->sdt_function);
731         sds = &server->sv_disp_stats1[sr->smb_com];
```



```
733         if (smb_mbc_decodef(&sr->command, "b", &sr->smb_wct) != 0) {
734                 disconnect = B_TRUE;
735                 goto report_error;
736         }

738         (void) MBC_SHADOW_CHAIN(&sr->smb_vwv, &sr->command,
739             sr->command.chain_offset, sr->smb_wct * 2);

741         if (smb_mbc_decodef(&sr->command, "#.w", sr->smb_wct*2, &sr->smb_bcc)) {
742                 disconnect = B_TRUE;
743                 goto report_error;
744         }

746         /*
747          * Ignore smb_bcc if CAP_LARGE_READX/CAP_LARGE_WRITEX
748          * and this is SmbReadX/SmbWriteX since this enables
749          * large reads/write and bcc is only 16-bits.
750          */
751         max_bytes = sr->command.max_bytes - sr->command.chain_offset;
752         if (sr->smb_com == SMB_COM_WRITE_ANDX) {
753                 /* Allow > BCC */
754                 byte_count = max_bytes;
755         } else if (max_bytes < (uint32_t)sr->smb_bcc) {
756                 /* BCC is bogus.  Will fail later. */
757                 byte_count = max_bytes;
758         } else {
759                 /* ordinary case */
760                 byte_count = (uint32_t)sr->smb_bcc;
761         }

763         /* Save these for kstat updates below. */
764         rxbytes = byte_count + 1 + 2 * sr->smb_wct;
765         txbase = sr->reply.chain_offset;

767         (void) MBC_SHADOW_CHAIN(&sr->smb_data, &sr->command,
768             sr->command.chain_offset, byte_count);

770         sr->command.chain_offset += byte_count;
771         if (sr->command.chain_offset > sr->command.max_bytes) {
772                 disconnect = B_TRUE;
773                 goto report_error;
774         }

776         /* Store pointers for later */
777         sr->cur_reply_offset = sr->reply.chain_offset;

779         if (is_andx_com(sr->smb_com)) {
780                 /* Peek ahead and don't disturb vwv */
781                 if (smb_mbc_peek(&sr->smb_vwv, sr->smb_vwv.chain_offset, "b.w",
782                     &sr->andx_com, &sr->andx_off) < 0) {
783                         disconnect = B_TRUE;
784                         goto report_error;
785                 }
786         } else {
787                 sr->andx_com = (unsigned char)-1;
788         }

790         mutex_enter(&sr->sr_mutex);
791         switch (sr->sr_state) {
792         case SMB_REQ_STATE_ACTIVE:
793                 break;
794         case SMB_REQ_STATE_CLEANED_UP:
795                 sr->sr_state = SMB_REQ_STATE_ACTIVE;
796                 break;
797         case SMB_REQ_STATE_CANCELLED:
```

```
798                 /*
799                  * Keep cancelled.  Handlers that might block will
800                  * check the state and return NT_STATUS_CANCELLED.
801                  */
802                 break;
803         default:
804                 ASSERT(0);
805                 break;
806         }
807         mutex_exit(&sr->sr_mutex);

809         /*
810          * Setup UID and TID information (if required). Both functions
811          * will set the sr credentials. In domain mode, the user and
812          * tree credentials should be the same. In share mode, the
813          * tree credentials (defined in the share definition) should
814          * override the user credentials.
815          */
816         if (!(sdd->sdt_flags & SDDF_SUPPRESS_UID) && (sr->uid_user == NULL)) {
817                 sr->uid_user = smb_session_lookup_uid(session, sr->smb_uid);
818                 if (sr->uid_user == NULL) {
819                         smbsr_error(sr, 0, ERRSRV, ERRbaduid);
820                         smbsr_cleanup(sr);
821                         goto report_error;
822                 }

824                 sr->user_cr = smb_user_getcred(sr->uid_user);
825         }
826         if (!(sdd->sdt_flags & SDDF_SUPPRESS_TID) && (sr->tid_tree == NULL)) {
827                 sr->tid_tree = smb_session_lookup_tree(session, sr->smb_tid);
828                 if (sr->tid_tree == NULL) {
829                         smbsr_error(sr, 0, ERRSRV, ERRinvnid);
830                         smbsr_cleanup(sr);
831                         goto report_error;
832                 }
833         }

835         sr->sr_time_start = gethrtime();
836         if ((sdrc = (*sdd->sdt_pre_op)(sr)) == SDRC_SUCCESS)
837                 sdrc = (*sdd->sdt_function)(sr);
838         (*sdd->sdt_post_op)(sr);

839         if (sdrc != SDRC_SR_KEPT) {
838                 (*sdd->sdt_post_op)(sr);
840                 smbsr_cleanup(sr);
841         }

843         /* Record latency and rx/tx bytes per:  server, session, share. */
844         {
845                 hrtime_t        dt;
846                 int64_t         rxb, txb;
847                 smb_disp_stats_t        *client_sds;    /* session */
848                 smb_disp_stats_t        *share_sds;  /* kshare */
849                 int     cmd_type;

851                 if (sdd->sdt_flags & SDDF_READOP) {
852                         cmd_type = SMBSRV_CLSH_READ;
853                 } else if (sdd->sdt_flags & SDDF_WRITEOP) {
854                         cmd_type = SMBSRV_CLSH_WRITE;
855                 } else {
856                         cmd_type = SMBSRV_CLSH_OTHER;
857                 }

859                 dt = gethrtime() - sr->sr_time_start;
860                 rxb = (int64_t)rxbytes;
861                 txb = (int64_t)(sr->reply.chain_offset - txbase);
```



```
863                 smb_latency_add_sample(&sds->sdt_lat, dt);
864                 atomic_add_64(&sds->sdt_rxb, rxb);
865                 atomic_add_64(&sds->sdt_txb, txb);

867                 client_sds = &session->s_stats[cmd_type];
868                 smb_latency_add_sample(&client_sds->sdt_lat, dt);
869                 atomic_add_64(&client_sds->sdt_rxb, rxb);
870                 atomic_add_64(&client_sds->sdt_txb, txb);

872                 if ((sr->tid_tree != NULL) &&
873                     (sr->tid_tree->t_kshare != NULL)) {
874                         share_sds =
875                             &sr->tid_tree->t_kshare->shr_stats[cmd_type];
876                         smb_latency_add_sample(&share_sds->sdt_lat, dt);
877                         atomic_add_64(&share_sds->sdt_rxb, rxb);
878                         atomic_add_64(&share_sds->sdt_txb, txb);
879                 }
880         }

882         switch (sdrc) {
883         case SDRC_SUCCESS:
884                 break;

886         case SDRC_DROP_VC:
887                 disconnect = B_TRUE;
888                 goto drop_connection;

890         case SDRC_NO_REPLY:
891                 /* will free sr */
892                 goto out;

894         case SDRC_SR_KEPT:
895                 /* Do NOT free sr */
896                 sr = NULL;
897                 goto out;

899         case SDRC_ERROR:
900                 goto report_error;

902         case SDRC_NOT_IMPLEMENTED:
903         default:
904                 smbsr_error(sr, 0, ERRDOS, ERRbadfunc);
905                 goto report_error;
906         }

908         /*
909          * If there's no AndX command, we're done.
910          */
911         if (sr->andx_com == 0xff)
912                 goto reply_ready;

914         /*
915          * Otherwise, we have to back-patch the AndXCommand and AndXOffset
916          * and continue processing.
917          */
918         sr->andx_prev_wct = sr->cur_reply_offset;
919         (void) smb_mbc_poke(&sr->reply, sr->andx_prev_wct + 1, "b.w",
920             sr->andx_com, MBC_LENGTH(&sr->reply));

922         sr->command.chain_offset = sr->orig_request_hdr + sr->andx_off;
923         sr->smb_com = sr->andx_com;
924         goto andx_more;

926 report_error:
927         sr->reply.chain_offset = sr->cur_reply_offset;
```

```
928         (void) smb_mbc_encodef(&sr->reply, "bw", 0, 0);

930         sr->smb_wct = 0;
931         sr->smb_bcc = 0;

933         if (sr->smb_rcls == 0 && sr->smb_reh == 0 && sr->smb_err == 0)
934                 smbsr_error(sr, 0, ERRSRV, ERRerror);

936 reply_ready:
937         smbsr_send_reply(sr);

939 drop_connection:
940         if (disconnect) {
941                 smb_rwx_rwenter(&session->s_lock, RW_WRITER);
942                 switch (session->s_state) {
943                 case SMB_SESSION_STATE_DISCONNECTED:
944                 case SMB_SESSION_STATE_TERMINATED:
945                         break;
946                 default:
947                         smb_soshutdown(session->sock);
948                         session->s_state = SMB_SESSION_STATE_DISCONNECTED;
949                         break;
950                 }
951                 smb_rwx_rwexit(&session->s_lock);
952         }

954 out:
955         if (sr != NULL) {
956                 mutex_enter(&sr->sr_mutex);
957                 sr->sr_state = SMB_REQ_STATE_COMPLETED;
958                 mutex_exit(&sr->sr_mutex);
959                 smb_request_free(sr);
960         }
961 }
```

*unchanged_portion_omitted_*

```
**********************************************************
   21989 Fri Mar 31 14:29:40 2017
new/usr/src/uts/common/fs/smbsrv/smb_negotiate.c
NEX-1643 dtrace provider for smbsrv
Also illumos 1841:
 DTrace smb provider was mis-implemented, doesn't exist.
Add back handlers for read/write raw, so that
 legacy dtrace consumers can find the probes.
Kill extra arg in smb_negotiate
Fix missing "done" probe with smb_notify
Add example consumer: smb-trace.d
fix soi_pid
**********************************************************
______unchanged_portion_omitted_

 348 smb_sdrc_t
 349 smb_pre_negotiate(smb_request_t *sr)
 350 {
 351         smb_kmod_cfg_t          *skc;
 352         smb_arg_negotiate_t     *negprot;
 353         int                     dialect;
 354         int                     pos;
 355         int                     rc = 0;

 357         skc = &sr->session->s_cfg;
 358         negprot = smb_srm_zalloc(sr, sizeof (smb_arg_negotiate_t));
 359         negprot->ni_index = -1;
 360         sr->sr_negprot = negprot;

 362         for (pos = 0; smbsr_decode_data_avail(sr); pos++) {
 363                 if (smbsr_decode_data(sr, "%L", sr, &negprot->ni_name) != 0) {
 364                         smbsr_error(sr, 0, ERRSRV, ERRerror);
 365                         rc = -1;
 366                         break;
 367                 }

 369                 if ((dialect = smb_xlate_dialect(negprot->ni_name)) < 0)
 370                         continue;

 372                 /*
 373                  * Conditionally recognize the SMB2 dialects.
 374                  */
 375                 if (dialect >= DIALECT_SMB2002 &&
 376                     skc->skc_max_protocol < SMB_VERS_2_BASE)
 377                         continue;

 379                 if (negprot->ni_dialect < dialect) {
 380                         negprot->ni_dialect = dialect;
 381                         negprot->ni_index = pos;
 382                 }
 383         }

 385         DTRACE_SMB_1(op__Negotiate__start, smb_request_t *, sr);
 385         DTRACE_SMB_2(op__Negotiate__start, smb_request_t *, sr,
 386             smb_arg_negotiate_t, negprot);

 387         return ((rc == 0) ? SDRC_SUCCESS : SDRC_ERROR);
 388 }

 390 void
 391 smb_post_negotiate(smb_request_t *sr)
 392 {
 393         smb_arg_negotiate_t     *negprot = sr->sr_negprot;

 395         DTRACE_SMB_1(op__Negotiate__done, smb_request_t *, sr);
 396         DTRACE_SMB_2(op__Negotiate__done, smb_request_t *, sr,
```



```
 397             smb_arg_negotiate_t, negprot);

 397         bzero(negprot, sizeof (smb_arg_negotiate_t));
 398 }
______unchanged_portion_omitted_
```

```
**********************************************************
   14314 Fri Mar 31 14:29:40 2017
new/usr/src/uts/common/fs/smbsrv/smb_nt_create_andx.c
Build provider 3rd arg from smb_request_t
hacking...
**********************************************************
/*
 * 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) 2007, 2010, Oracle and/or its affiliates. All rights reserved.
 * Copyright 2016 Nexenta Systems, Inc.  All rights reserved.
 */

/*
 * This command is used to create or open a file or directory.
 */


#include <smbsrv/smb_kproto.h>
#include <smbsrv/smb_fsops.h>
#include <smbsrv/smb_vops.h>

int smb_nt_create_enable_extended_response = 1;

/*
 * smb_com_nt_create_andx
 *
 * This command is used to create or open a file or directory.
 *
 *  Client Request                      Description
 *  ================================    ================================
 *
 *  UCHAR WordCount;                    Count of parameter words = 24
 *  UCHAR AndXCommand;                  Secondary command;  0xFF = None
 *  UCHAR AndXReserved;                 Reserved (must be 0)
 *  USHORT AndXOffset;                  Offset to next command WordCount
 *  UCHAR Reserved;                     Reserved (must be 0)
 *  USHORT NameLength;                  Length of Name[] in bytes
 *  ULONG Flags;                        Create bit set:
 *                                      0x02 - Request an oplock
 *                                      0x04 - Request a batch oplock
 *                                      0x08 - Target of open must be
 *                                      directory
 *  ULONG RootDirectoryFid;             If non-zero, open is relative to
 *                                      this directory
 *  ACCESS_MASK DesiredAccess;          access desired
 *  LARGE_INTEGER AllocationSize;       Initial allocation size
 *  ULONG ExtFileAttributes;            File attributes
```



```
 *  ULONG ShareAccess;                  Type of share access
 *  ULONG CreateDisposition;            Action to take if file exists or
 *                                      not
 *  ULONG CreateOptions;                Options to use if creating a file
 *  ULONG ImpersonationLevel;           Security QOS information
 *  UCHAR SecurityFlags;                Security tracking mode flags:
 *                                      0x1 - SECURITY_CONTEXT_TRACKING
 *                                      0x2 - SECURITY_EFFECTIVE_ONLY
 *  USHORT ByteCount;                   Length of byte parameters
 *  STRING Name[];                      File to open or create
 *
 * The DesiredAccess parameter is specified in section 3.7 on  Access Mask
 * Encoding.
 *
 * If no value is specified, it still allows an application to query
 * attributes without actually accessing the file.
 *
 * The ExtFIleAttributes parameter specifies the file attributes and flags
 * for the file. The parameter's value is the sum of allowed attributes and
 * flags defined in section 3.11 on  Extended File Attribute Encoding
 *
 * The ShareAccess field Specifies how this file can be shared. This
 * parameter must be some combination of the following values:
 *
 * Name              Value      Meaning
 *                   0          Prevents the file from being shared.
 * FILE_SHARE_READ   0x00000001 Other open operations can be performed on
 *                               the file for read access.
 * FILE_SHARE_WRITE  0x00000002 Other open operations can be performed on
 *                               the file for write access.
 * FILE_SHARE_DELETE 0x00000004 Other open operations can be performed on
 *                               the file for delete access.
 *
 * The CreateDisposition parameter can contain one of the following values:
 *
 * CREATE_NEW        Creates a new file. The function fails if the
 *                   specified file already exists.
 * CREATE_ALWAYS     Creates a new file. The function overwrites the file
 *                   if it exists.
 * OPEN_EXISTING     Opens the file. The function fails if the file does
 *                   not exist.
 * OPEN_ALWAYS       Opens the file, if it exists. If the file does not
 *                   exist, act like CREATE_NEW.
 * TRUNCATE_EXISTING Opens the file. Once opened, the file is truncated so
 *                   that its size is zero bytes. The calling process must
 *                   open the file with at least GENERIC_WRITE access. The
 *                   function fails if the file does not exist.
 *
 * The ImpersonationLevel parameter can contain one or more of the
 * following values:
 *
 * SECURITY_ANONYMOUS        Specifies to impersonate the client at the
 *                           Anonymous impersonation level.
 * SECURITY_IDENTIFICATION   Specifies to impersonate the client at the
 *                           Identification impersonation level.
 * SECURITY_IMPERSONATION    Specifies to impersonate the client at the
 *                           Impersonation impersonation level.
 * SECURITY_DELEGATION       Specifies to impersonate the client at the
 *                           Delegation impersonation level.
 *
 * The SecurityFlags parameter can have either of the following two flags
 * set:
 *
 * SECURITY_CONTEXT_TRACKING  Specifies that the security tracking mode is
 *                            dynamic. If this flag is not specified,
 *                            Security Tracking Mode is static.
```

```
 127  * SECURITY_EFFECTIVE_ONLY    Specifies that only the enabled aspects of
 128  *                            the client's security context are available
 129  *                            to the server. If you do not specify this
 130  *                            flag, all aspects of the client's security
 131  *                            context are available. This flag allows the
 132  *                            client to limit the groups and privileges
 133  *                            that a server can use while impersonating the
 134  *                            client.
 135  *
 136  * The response is as follows:
 137  *
 138  *  Server Response                  Description
 139  *  ================================ ================================
 140  *
 141  *  UCHAR WordCount;                 Count of parameter words = 26
 142  *  UCHAR AndXCommand;  Secondary    0xFF = None
 143  *  command;
 144  *  UCHAR AndXReserved;              MBZ
 145  *  USHORT AndXOffset;               Offset to next command WordCount
 146  *  UCHAR OplockLevel;               The oplock level granted
 147  *                                   0 - No oplock granted
 148  *                                   1 - Exclusive oplock granted
 149  *                                   2 - Batch oplock granted
 150  *                                   3 - Level II oplock granted
 151  *  USHORT Fid;                      The file ID
 152  *  ULONG CreateAction;              The action taken
 153  *  TIME CreationTime;               The time the file was created
 154  *  TIME LastAccessTime;             The time the file was accessed
 155  *  TIME LastWriteTime;              The time the file was last written
 156  *  TIME ChangeTime;                 The time the file was last changed
 157  *  ULONG ExtFileAttributes;         The file attributes
 158  *  LARGE_INTEGER AllocationSize;    The number of bytes allocated
 159  *  LARGE_INTEGER EndOfFile;         The end of file offset
 160  *  USHORT FileType;
 161  *  USHORT DeviceState;              state of IPC device (e.g. pipe)
 162  *  BOOLEAN Directory;               TRUE if this is a directory
 163  *  USHORT ByteCount;                = 0
 164  *
 165  * The following SMBs may follow SMB_COM_NT_CREATE_ANDX:
 166  *
 167  *    SMB_COM_READ    SMB_COM_READ_ANDX
 168  *    SMB_COM_IOCTL
 169  */
 170 smb_sdrc_t
 171 smb_pre_nt_create_andx(smb_request_t *sr)
 172 {
 173         struct open_param *op = &sr->arg.open;
 174         uint8_t SecurityFlags;
 175         uint32_t ImpersonationLevel;
 176         uint16_t NameLength;
 177         int rc;

 179         bzero(op, sizeof (sr->arg.open));

 181         rc = smbsr_decode_vwv(sr, "5.wlllqllllb",
 182             &NameLength,
 183             &op->nt_flags,
 184             &op->rootdirfid,
 185             &op->desired_access,
 186             &op->dsize,
 187             &op->dattr,
 188             &op->share_access,
 189             &op->create_disposition,
 190             &op->create_options,
 191             &ImpersonationLevel,
 192             &SecurityFlags);
```



```
 194         if (rc == 0) {
 195                 if (NameLength == 0) {
 196                         op->fqi.fq_path.pn_path = "\\";
 197                 } else if (NameLength >= SMB_MAXPATHLEN) {
 198                         smbsr_error(sr, NT_STATUS_OBJECT_NAME_INVALID,
 199                             ERRDOS, ERROR_PATH_NOT_FOUND);
 200                         rc = -1;
 201                 } else {
 202                         rc = smbsr_decode_data(sr, "%#u", sr, NameLength,
 203                             &op->fqi.fq_path.pn_path);
 204                 }
 205         }

 207         op->op_oplock_level = SMB_OPLOCK_NONE;
 208         if (op->nt_flags & NT_CREATE_FLAG_REQUEST_OPLOCK) {
 209                 if (op->nt_flags & NT_CREATE_FLAG_REQUEST_OPBATCH)
 210                         op->op_oplock_level = SMB_OPLOCK_BATCH;
 211                 else
 212                         op->op_oplock_level = SMB_OPLOCK_EXCLUSIVE;
 213         }

 215         DTRACE_SMB_1(op__NtCreateX__start, smb_request_t *, sr); /* arg.open */
 215         DTRACE_SMB_2(op__NtCreateX__start, smb_request_t *, sr,
 216             struct open_param *, op);

 217         return ((rc == 0) ? SDRC_SUCCESS : SDRC_ERROR);
 218 }
______unchanged_portion_omitted_
```

```
**********************************************************
    8083 Fri Mar 31 14:29:41 2017
new/usr/src/uts/common/fs/smbsrv/smb_nt_transact_create.c
Build provider 3rd arg from smb_request_t
hacking...
NEX-1643 dtrace provider for smbsrv
Also illumos 1841:
 DTrace smb provider was mis-implemented, doesn't exist.
Add back handlers for read/write raw, so that
 legacy dtrace consumers can find the probes.
Kill extra arg in smb_negotiate
Fix missing "done" probe with smb_notify
Add example consumer: smb-trace.d
fix soi_pid
**********************************************************
```

```
/*
 * 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) 2007, 2010, Oracle and/or its affiliates. All rights reserved.
 * Copyright 2016 Nexenta Systems, Inc.  All rights reserved.
 */

/*
 * This command is used to create or open a file or directory, when EAs
 * or an SD must be applied to the file. The functionality is similar
 * to SmbNtCreateAndx with the option to supply extended attributes or
 * a security descriptor.
 *
 * Note: we don't decode the extended attributes because we don't
 * support them at this time.
 */

#include <smbsrv/smb_kproto.h>
#include <smbsrv/smb_fsops.h>

extern int smb_nt_create_enable_extended_response;

/*
 * smb_nt_transact_create
 *
 * This command is used to create or open a file or directory, when EAs
 * or an SD must be applied to the file. The request parameter block
 * encoding, data block encoding and output parameter block encoding are
 * described in CIFS section 4.2.2.
 *
 * The format of the command is SmbNtTransact but it is basically the same
 * as SmbNtCreateAndx with the option to supply extended attributes or a
 * security descriptor. For information not defined in CIFS section 4.2.2
```



```
 * see section 4.2.1 (NT_CREATE_ANDX).
 */
smb_sdrc_t
smb_pre_nt_transact_create(smb_request_t *sr, smb_xa_t *xa)
{
        struct open_param *op = &sr->arg.open;
        uint8_t SecurityFlags;
        uint32_t EaLength;
        uint32_t ImpersonationLevel;
        uint32_t NameLength;
        uint32_t sd_len;
        uint32_t status;
        smb_sd_t sd;
        int rc;

        bzero(op, sizeof (sr->arg.open));

        rc = smb_mbc_decodef(&xa->req_param_mb, "%lllqlllllllb",
            sr,
            &op->nt_flags,
            &op->rootdirfid,
            &op->desired_access,
            &op->dsize,
            &op->dattr,
            &op->share_access,
            &op->create_disposition,
            &op->create_options,
            &sd_len,
            &EaLength,
            &NameLength,
            &ImpersonationLevel,
            &SecurityFlags);

        if (rc == 0) {
                if (NameLength == 0) {
                        op->fqi.fq_path.pn_path = "\\";
                } else if (NameLength >= SMB_MAXPATHLEN) {
                        smbsr_error(sr, NT_STATUS_OBJECT_NAME_INVALID,
                            ERRDOS, ERROR_INVALID_NAME);
                        rc = -1;
                } else {
                        rc = smb_mbc_decodef(&xa->req_param_mb, "%#u",
                            sr, NameLength, &op->fqi.fq_path.pn_path);
                }
        }

        op->op_oplock_level = SMB_OPLOCK_NONE;
        if (op->nt_flags & NT_CREATE_FLAG_REQUEST_OPLOCK) {
                if (op->nt_flags & NT_CREATE_FLAG_REQUEST_OPBATCH)
                        op->op_oplock_level = SMB_OPLOCK_BATCH;
                else
                        op->op_oplock_level = SMB_OPLOCK_EXCLUSIVE;
        }

        if (sd_len) {
                status = smb_decode_sd(&xa->req_data_mb, &sd);
                if (status != NT_STATUS_SUCCESS) {
                        smbsr_error(sr, status, 0, 0);
                        return (SDRC_ERROR);
                }
                op->sd = kmem_alloc(sizeof (smb_sd_t), KM_SLEEP);
                *op->sd = sd;
        } else {
                op->sd = NULL;
        }
```

```
118         DTRACE_SMB_1(op__NtTransactCreate__start, smb_request_t *, sr); /* arg.o
118         DTRACE_SMB_2(op__NtTransactCreate__start, smb_request_t *, sr,
119             struct open_param *, op);

120         return ((rc == 0) ? SDRC_SUCCESS : SDRC_ERROR);
121 }

123 void
124 smb_post_nt_transact_create(smb_request_t *sr, smb_xa_t *xa)
125 {
126         smb_sd_t *sd = sr->arg.open.sd;
127         _NOTE(ARGUNUSED(xa))

129         DTRACE_SMB_1(op__NtTransactCreate__done, smb_request_t *, sr);
129         DTRACE_SMB_2(op__NtTransactCreate__done, smb_request_t *, sr,
130             smb_xa_t *, xa);

131         if (sd) {
132                 smb_sd_term(sd);
133                 kmem_free(sd, sizeof (smb_sd_t));
134         }

136         if (sr->arg.open.dir != NULL) {
137                 smb_ofile_release(sr->arg.open.dir);
138                 sr->arg.open.dir = NULL;
139         }
140 }
```

*unchanged_portion_omitted_*

```
**********************************************************
    7314 Fri Mar 31 14:29:41 2017
new/usr/src/uts/common/fs/smbsrv/smb_nt_transact_notify_change.c
NEX-1643 dtrace provider for smbsrv
Also illumos 1841:
 DTrace smb provider was mis-implemented, doesn't exist.
Add back handlers for read/write raw, so that
 legacy dtrace consumers can find the probes.
Kill extra arg in smb_negotiate
Fix missing "done" probe with smb_notify
Add example consumer: smb-trace.d
fix soi_pid
**********************************************************
______unchanged_portion_omitted_
```

```
143 /*
144  * This is called via taskq_dispatch in smb_notify.c
145  * to finish up an NT transact notify change request.
146  */
147 void
148 smb_nt_transact_notify_finish(void *arg)
149 {
150         smb_request_t   *sr = arg;
151         struct smb_xa   *xa;
152         smb_disp_stats_t *sds;
153         int             total_bytes, n_setup, n_param, n_data;
154         int             param_off, param_pad, data_off, data_pad;
155         uint32_t        status;

157         SMB_REQ_VALID(sr);

159         /*
160          * Common part of notify, puts data in sr->raw_data
161          */
162         status = smb_notify_act3(sr);

164         /*
165          * SMB1 expects an empty trans response after the
166          * FID we're watching is closed.
167          */
168         if (status == NT_STATUS_NOTIFY_CLEANUP) {
169                 status = 0;
170                 MBC_FLUSH(&sr->raw_data);
171         }

173         if (status != 0) {
174                 smbsr_status(sr, status, 0, 0);
175                 if (NT_SC_SEVERITY(status) == NT_STATUS_SEVERITY_ERROR) {
176                         (void) smb_mbc_encodef(&sr->reply, "bwbw",
177                             (short)0, 0L, (short)0, 0L);
178                         goto sendit;
179                 }
180                 /* Else continue with NT_STATUS_NOTIFY_ENUM_DIR etc. */
181         }

183         /*
184          * setup the NT transact reply
185          *
186          * Note that this is a copy/paste of code from
187          * smb_nt_trans_dispatch(), with minor changes.
188          * Intentionally keeping this similar to the
189          * original rather than hand-optimizing.
190          *
191          * The "setup" and "data" parts of this trans reply
192          * (n_setup, n_data, rep_setup_mb, rep_data_mb) are
193          * always empty.  sr->raw_data replaces rep_param_mb.
```



```
194          */
195         xa = sr->r_xa;
196         n_setup = MBC_LENGTH(&xa->rep_setup_mb);
197         n_param = MBC_LENGTH(&sr->raw_data);
198         n_data  = MBC_LENGTH(&xa->rep_data_mb);

200         n_setup = (n_setup + 1) / 2;    /* Convert to setup words */
201         param_pad = 1;                  /* must be one */
202         param_off = param_pad + 32 + 37 + (n_setup << 1) + 2;
203         /* Pad to 4 bytes */
204         data_pad = (4 - ((param_off + n_param) & 3)) % 4;
205         /* Param off from hdr */
206         data_off = param_off + n_param + data_pad;
207         total_bytes = param_pad + n_param + data_pad + n_data;

209         (void) smbsr_encode_result(sr, 18+n_setup, total_bytes,
210             "b3.llllllllbCw#.C#.C",
211             18 + n_setup,       /* wct */
212             n_param,            /* Total Parameter Bytes */
213             n_data,             /* Total Data Bytes */
214             n_param,            /* Total Parameter Bytes this buffer */
215             param_off,          /* Param offset from header start */
216             0,                  /* Param displacement */
217             n_data,             /* Total Data Bytes this buffer */
218             data_off,           /* Data offset from header start */
219             0,                  /* Data displacement */
220             n_setup,            /* suwcnt */
221             &xa->rep_setup_mb,  /* setup[] */
222             total_bytes,        /* Total data bytes */
223             param_pad,
224             &sr->raw_data,      /* output mbc */
225             data_pad,
226             &xa->rep_data_mb);

228 sendit:
229         DTRACE_SMB_1(op__NtTransactNotify__done2, smb_request_t *, sr);

231         sds = &sr->sr_server->sv_disp_stats1[sr->smb_com];
232         atomic_add_64(&sds->sdt_txb, (int64_t)sr->reply.chain_offset);

234         smbsr_send_reply(sr);   /* also puts the SMB header. */
235         smbsr_cleanup(sr);

237         mutex_enter(&sr->sr_mutex);
238         sr->sr_state = SMB_REQ_STATE_COMPLETED;
239         mutex_exit(&sr->sr_mutex);

241         smb_request_free(sr);
242 }
```

______unchanged_portion_omitted_

```
**********************************************************
   20246 Fri Mar 31 14:29:41 2017
new/usr/src/uts/common/fs/smbsrv/smb_open_andx.c
Build provider 3rd arg from smb_request_t
hacking...
**********************************************************
```

```
/*
 * 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) 2007, 2010, Oracle and/or its affiliates. All rights reserved.
 * Copyright 2016 Nexenta Systems, Inc.  All rights reserved.
 */

#include <smbsrv/smb_kproto.h>
#include <smbsrv/smb_fsops.h>
#include <smbsrv/smb_vops.h>

int smb_open_dsize_check = 0;

/*
 *  Client Request                  Description
 *  ================================ ================================
 *
 *  UCHAR WordCount;                Count of parameter words = 15
 *  UCHAR AndXCommand;              Secondary (X) command;  0xFF =
 *                                   none
 *  UCHAR AndXReserved;             Reserved (must be 0)
 *  USHORT AndXOffset;              Offset to next command WordCount
 *  USHORT Flags;                   Additional information: bit set-
 *                                   0 - return additional info
 *                                   1 - exclusive oplock requested
 *                                   2 - batch oplock requested
 *  USHORT DesiredAccess;           File open mode
 *  USHORT SearchAttributes;
 *  USHORT FileAttributes;
 *  UTIME CreationTime;             Creation timestamp for file if it
 *                                   gets created
 *  USHORT OpenFunction;            Action to take if file exists
 *  ULONG AllocationSize;           Bytes to reserve on create or
 *                                   truncate
 *  ULONG Reserved[2];              Must be 0
 *  USHORT ByteCount;               Count of data bytes;    min = 1
 *  UCHAR BufferFormat              0x04
 *  STRING FileName;
 *
 *  Server Response                 Description
 *  ================================ ================================
 *
 *  UCHAR WordCount;                Count of parameter words = 15
 *  UCHAR AndXCommand;              Secondary (X) command;  0xFF =
 *                                   none
 *  UCHAR AndXReserved;             Reserved (must be 0)
 *  USHORT AndXOffset;              Offset to next command WordCount
 *  USHORT Fid;                     File handle
 *  USHORT FileAttributes;
 *  UTIME LastWriteTime;
 *  ULONG DataSize;                 Current file size
 *  USHORT GrantedAccess;           Access permissions actually
 *                                   allowed
 *  USHORT FileType;                Type of file opened
 *  USHORT DeviceState;             State of the named pipe
 *  USHORT Action;                  Action taken
 *  ULONG ServerFid;                Server unique file id
 *  USHORT Reserved;                Reserved (must be 0)
 *  USHORT ByteCount;               Count of data bytes = 0
 *
 * DesiredAccess describes the access the client desires for the file (see
 * section 3.6 -  Access Mode Encoding).
 *
 * OpenFunction specifies the action to be taken depending on whether or
 * not the file exists (see section 3.8 -  Open Function Encoding).  Action
 *
 * in the response specifies the action as a result of the Open request
 * (see section 3.9 -  Open Action Encoding).
 *
 * SearchAttributes indicates the attributes that the file must have to be
 * found while searching to see if it exists.  The encoding of this field
 * is described in the "File Attribute Encoding" section elsewhere in this
 * document.  If SearchAttributes is zero then only normal files are
 * returned.  If the system file, hidden or directory attributes are
 * specified then the search is inclusive -- both the specified type(s) of
 * files and normal files are returned.
 *
 * FileType returns the kind of resource actually opened:
 *
 *  Name                        Value  Description
 *  ========================== ====== ================================
 *
 *  FileTypeDisk                0      Disk file or directory as defined
 *                                      in the attribute field
 *  FileTypeByteModePipe        1      Named pipe in byte mode
 *  FileTypeMessageModePipe     2      Named pipe in message mode
 *  FileTypePrinter             3      Spooled printer
 *  FileTypeUnknown             0xFFFF Unrecognized resource type
 *
 * If bit0 of Flags is clear, the FileAttributes, LastWriteTime, DataSize,
 * FileType, and DeviceState have indeterminate values in the response.
 *
 * This SMB can request an oplock on the opened file.  Oplocks are fully
 * described in the "Oplocks" section elsewhere in this document, and there
 * is also discussion of oplocks in the SMB_COM_LOCKING_ANDX SMB
 * description.  Bit1 and bit2 of the Flags field are used to request
 * oplocks during open.
 *
 * The following SMBs may follow SMB_COM_OPEN_ANDX:
 *
 *    SMB_COM_READ    SMB_COM_READ_ANDX
 *    SMB_COM_IOCTL
 */

/*
 * This message is sent to obtain a file handle for a data file.  This
 * returned Fid is used in subsequent client requests such as read, write,
 * close, etc.
```

```
 127  *
 128  * Client Request                     Description
 129  * ================================== ==================================
 130  *
 131  * UCHAR WordCount;                   Count of parameter words = 2
 132  * USHORT DesiredAccess;              Mode - read/write/share
 133  * USHORT SearchAttributes;
 134  * USHORT ByteCount;                  Count of data bytes;    min = 2
 135  * UCHAR BufferFormat;                0x04
 136  * STRING FileName[];                 File name
 137  *
 138  * FileName is the fully qualified file name, relative to the root of the
 139  * share specified in the Tid field of the SMB header.  If Tid in the SMB
 140  * header refers to a print share, this SMB creates a new file which will
 141  * be spooled to the printer when closed.  In this case, FileName is
 142  * ignored.
 143  *
 144  * SearchAttributes specifies the type of file desired.  The encoding is
 145  * described in the "File Attribute Encoding" section.
 146  *
 147  * DesiredAccess controls the mode under which the file is opened, and the
 148  * file will be opened only if the client has the appropriate permissions.
 149  * The encoding of DesiredAccess is discussed in the section entitled
 150  * "Access Mode Encoding".
 151  *
 152  * Server Response                    Description
 153  * ================================== ==================================
 154  *
 155  * UCHAR WordCount;                   Count of parameter words = 7
 156  * USHORT Fid;                        File handle
 157  * USHORT FileAttributes;             Attributes of opened file
 158  * UTIME LastWriteTime;               Time file was last written
 159  * ULONG DataSize;                    File size
 160  * USHORT GrantedAccess;              Access allowed
 161  * USHORT ByteCount;                  Count of data bytes = 0
 162  *
 163  * Fid is the handle value which should be used for subsequent file
 164  * operations.
 165  *
 166  * FileAttributes specifies the type of file obtained.  The encoding is
 167  * described in the "File Attribute Encoding" section.
 168  *
 169  * GrantedAccess indicates the access permissions actually allowed, and may
 170  * have one of the following values:
 171  *
 172  *    0  read-only
 173  *    1  write-only
 174  *    2 read/write
 175  *
 176  * File Handles (Fids) are scoped per client.  A Pid may reference any Fid
 177  * established by itself or any other Pid on the client (so far as the
 178  * server is concerned).  The actual accesses allowed through the Fid
 179  * depends on the open and deny modes specified when the file was opened
 180  * (see below).
 181  *
 182  * The MS-DOS compatibility mode of file open provides exclusion at the
 183  * client level.  A file open in compatibility mode may be opened (also in
 184  * compatibility mode) any number of times for any combination of reading
 185  * and writing (subject to the user's permissions) by any Pid on the same
 186  * client.  If the first client has the file open for writing, then the
 187  * file may not be opened in any way by any other client.  If the first
 188  * client has the file open only for reading, then other clients may open
 189  * the file, in compatibility mode, for reading..  The above
 190  * notwithstanding, if the filename has an extension of .EXE, .DLL, .SYM,
 191  * or .COM other clients are permitted to open the file regardless of
 192  * read/write open modes of other compatibility mode opens.  However, once
```



```
 193  * multiple clients have the file open for reading, no client is permitted
 194  * to open the file for writing and no other client may open the file in
 195  * any mode other than compatibility mode.
 196  *
 197  * The other file exclusion modes (Deny read/write, Deny write, Deny read,
 198  * Deny none) provide exclusion at the file level.  A file opened in any
 199  * "Deny" mode may be opened again only for the accesses allowed by the
 200  * Deny mode (subject to the user's permissions).  This is true regardless
 201  * of the identity of the second opener -a different client, a Pid from the
 202  * same client, or the Pid that already has the file open.  For example, if
 203  * a file is open in "Deny write" mode a second open may only obtain read
 204  * permission to the file.
 205  *
 206  * Although Fids are available to all Pids on a client, Pids other than the
 207  * owner may not have the full access rights specified in the open mode by
 208  * the Fid's creator.  If the open creating the Fid specified a deny mode,
 209  * then any Pid using the Fid, other than the creating Pid, will have only
 210  * those access rights determined by "anding" the open mode rights and the
 211  * deny mode rights, i.e., the deny mode is checked on all file accesses.
 212  * For example, if a file is opened for Read/Write in Deny write mode, then
 213  * other clients may only read the file and cannot write; if a file is
 214  * opened for Read in Deny read mode, then the other clients can neither
 215  * read nor write the file.
 216  */

 218 smb_sdrc_t
 219 smb_pre_open(smb_request_t *sr)
 220 {
 221         struct open_param *op = &sr->arg.open;
 222         int rc;

 224         bzero(op, sizeof (sr->arg.open));

 226         rc = smbsr_decode_vwv(sr, "ww", &op->omode, &op->fqi.fq_sattr);
 227         if (rc == 0)
 228                 rc = smbsr_decode_data(sr, "%S", sr, &op->fqi.fq_path.pn_path);

 230         DTRACE_SMB_1(op__Open__start, smb_request_t *, sr); /* arg.open */
 230         DTRACE_SMB_2(op__Open__start, smb_request_t *, sr,
 231             struct open_param *, op);

 232         return ((rc == 0) ? SDRC_SUCCESS : SDRC_ERROR);
 233 }
_____unchanged_portion_omitted_

 319 int smb_openx_enable_extended_response = 1;

 321 /*
 322  * smb_pre_open_andx
 323  * For compatibility with windows servers, the search attributes
 324  * specified in the request are ignored.
 325  */
 326 smb_sdrc_t
 327 smb_pre_open_andx(smb_request_t *sr)
 328 {
 329         struct open_param *op = &sr->arg.open;
 330         uint16_t openx_flags;
 331         uint32_t alloc_size;
 332         uint32_t creation_time;
 333         uint16_t file_attr, sattr;
 334         int rc;

 336         bzero(op, sizeof (sr->arg.open));

 338         rc = smbsr_decode_vwv(sr, "b.wwwwwlwll4.", &sr->andx_com,
 339             &sr->andx_off, &openx_flags, &op->omode, &sattr,
```

```
340             &file_attr, &creation_time, &op->ofun, &alloc_size, &op->timeo);

342         if (rc == 0) {
343                 rc = smbsr_decode_data(sr, "%u", sr, &op->fqi.fq_path.pn_path);

345                 op->dattr = file_attr;
346                 op->dsize = alloc_size;

348                 /*
349                  * The openx_flags use some "extended" flags that
350                  * happen to match some of the NtCreateX flags.
351                  */
352                 if (openx_flags & NT_CREATE_FLAG_REQUEST_OPLOCK)
353                         op->op_oplock_level = SMB_OPLOCK_EXCLUSIVE;
354                 else if (openx_flags & NT_CREATE_FLAG_REQUEST_OPBATCH)
355                         op->op_oplock_level = SMB_OPLOCK_BATCH;
356                 else
357                         op->op_oplock_level = SMB_OPLOCK_NONE;
358                 if (openx_flags & NT_CREATE_FLAG_EXTENDED_RESPONSE)
359                         op->nt_flags |= NT_CREATE_FLAG_EXTENDED_RESPONSE;

361                 if ((creation_time != 0) && (creation_time != UINT_MAX))
362                         op->crtime.tv_sec =
363                             smb_time_local_to_gmt(sr, creation_time);
364                 op->crtime.tv_nsec = 0;

366                 op->create_disposition = smb_ofun_to_crdisposition(op->ofun);
367         }

369         DTRACE_SMB_1(op__OpenX__start, smb_request_t *, sr); /* arg.open */
370         DTRACE_SMB_2(op__OpenX__start, smb_request_t *, sr,
371             struct open_param *, op);

371         return ((rc == 0) ? SDRC_SUCCESS : SDRC_ERROR);
372 }
```

*unchanged_portion_omitted_*

```
**********************************************************
    8848 Fri Mar 31 14:29:42 2017
new/usr/src/uts/common/fs/smbsrv/smb_print.c
Build provider 3rd arg from smb_request_t
hacking...
**********************************************************
```

```
/*
 * 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) 2007, 2010, Oracle and/or its affiliates. All rights reserved.
 * Copyright 2013 Nexenta Systems, Inc.  All rights reserved.
 */

/*
 * SMB print interface.
 */

#include <smbsrv/smb_kproto.h>
#include <sys/unistd.h>
#include <sys/stat.h>
#include <sys/types.h>
#include <sys/fcntl.h>
#include <smbsrv/smb_share.h>

/*
 * Starts the creation of a new printer file, which will be deleted
 * automatically once it has been closed and printed.
 *
 * SetupLength is the number of bytes in the first part of the resulting
 * print spool file which contains printer-specific control strings.
 *
 * Mode can have the following values:
 *      0     Text mode.  The server may optionally
 *            expand tabs to a series of spaces.
 *      1     Graphics mode.  No conversion of data
 *            should be done by the server.
 *
 * IdentifierString can be used by the server to provide some sort of
 * per-client identifying component to the print file.
 *
 * When the file is closed, it will be sent to the spooler and printed.
 */
smb_sdrc_t
smb_pre_open_print_file(smb_request_t *sr)
{
        struct open_param       *op = &sr->arg.open;
        char                    *path;
        char                    *identifier;
```



```
        uint32_t                new_id;
        uint16_t                setup;
        uint16_t                mode;
        int                     rc;
        static uint32_t         tmp_id = 10000;

        bzero(op, sizeof (sr->arg.open));
        rc = smbsr_decode_vwv(sr, "ww", &setup, &mode);
        if (rc == 0)
                rc = smbsr_decode_data(sr, "%S", sr, &identifier);

        if (rc == 0) {
                path = smb_srm_zalloc(sr, MAXPATHLEN);
                op->fqi.fq_path.pn_path = path;
                new_id = atomic_inc_32_nv(&tmp_id);
                (void) snprintf(path, MAXPATHLEN, "%s%05u", identifier, new_id);
        }

        op->create_disposition = FILE_OVERWRITE_IF;
        op->create_options = FILE_NON_DIRECTORY_FILE;
        DTRACE_SMB_1(op__OpenPrintFile__start, smb_request_t *, sr); /* arg.open
        DTRACE_SMB_2(op__OpenPrintFile__start, smb_request_t *, sr,
            struct open_param *, op);

        return ((rc == 0) ? SDRC_SUCCESS : SDRC_ERROR);
}
```

_____unchanged_portion_omitted_

****
27492 Fri Mar 31 14:29:42 2017
new/usr/src/uts/common/fs/smbsrv/smb_query_fileinfo.c
Build provider 3rd arg from smb_request_t
hacking...
NEX-1643 dtrace provider for smbsrv
Also illumos 1841:
DTrace smb provider was mis-implemented, doesn't exist.
Add back handlers for read/write raw, so that
legacy dtrace consumers can find the probes.
Kill extra arg in smb_negotiate
Fix missing "done" probe with smb_notify
Add example consumer: smb-trace.d
fix soi_pid
****
_____***unchanged_portion_omitted_***

```
 141 /*
 142  * smb_com_query_information (aka getattr)
 143  */
 144 smb_sdrc_t
 145 smb_pre_query_information(smb_request_t *sr)
 146 {
 147         int rc;
 148         smb_fqi_t *fqi = &sr->arg.dirop.fqi;

 150         rc = smbsr_decode_data(sr, "%S", sr, &fqi->fq_path.pn_path);

 152         DTRACE_SMB_1(op__QueryInformation__start, smb_request_t *, sr); /* arg.d
 152         DTRACE_SMB_2(op__QueryInformation__start, smb_request_t *, sr,
 153             smb_fqi_t *, fqi);

 154         return ((rc == 0) ? SDRC_SUCCESS : SDRC_ERROR);
 155 }
```

_____***unchanged_portion_omitted_***

```
**********************************************************
   14415 Fri Mar 31 14:29:42 2017
new/usr/src/uts/common/fs/smbsrv/smb_read.c
Build provider 3rd arg from smb_request_t
hacking...
NEX-1643 dtrace provider for smbsrv
Also illumos 1841:
 DTrace smb provider was mis-implemented, doesn't exist.
Add back handlers for read/write raw, so that
 legacy dtrace consumers can find the probes.
Kill extra arg in smb_negotiate
Fix missing "done" probe with smb_notify
Add example consumer: smb-trace.d
fix soi_pid
**********************************************************
```

```
   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) 2007, 2010, Oracle and/or its affiliates. All rights reserved.
  23  * Copyright 2016 Nexenta Systems, Inc.  All rights reserved.
  24  */

  26 #include <smbsrv/smb_kproto.h>
  27 #include <smbsrv/smb_fsops.h>

  29 /*
  30  * The maximum number of bytes to return from SMB Core
  31  * SmbRead or SmbLockAndRead.
  32  */
  33 #define SMB_CORE_READ_MAX       4432

  35 /*
  36  * The limit in bytes for SmbReadX.
  37  */
  38 #define SMB_READX_MAX           0x10000

  40 int smb_common_read(smb_request_t *, smb_rw_param_t *);

  42 /*
  43  * Read bytes from a file or named pipe (SMB Core).
  44  *
  45  * The requested count specifies the number of bytes desired.  Offset
  46  * is limited to 32 bits, so this client request is inappropriate for
  47  * files with 64 bit offsets.
  48  *
  49  * On return, count is the number of bytes actually being returned, which
  50  * may be less than the count requested only if a read specifies bytes
  51  * beyond the current file size.  In this case only the bytes that exist
```



```
  52  * are returned.  A read completely beyond the end of file results in a
  53  * response of length zero.  This is the only circumstance when a zero
  54  * length response is generated.  A count returned which is less than the
  55  * count requested is the end of file indicator.
  56  */
  57 smb_sdrc_t
  58 smb_pre_read(smb_request_t *sr)
  59 {
  60         smb_rw_param_t *param;
  61         uint32_t off_low;
  62         uint16_t count;
  63         uint16_t remcnt;
  64         int rc;

  66         param = kmem_zalloc(sizeof (smb_rw_param_t), KM_SLEEP);
  67         sr->arg.rw = param;

  69         rc = smbsr_decode_vwv(sr, "wwlw", &sr->smb_fid,
  70             &count, &off_low, &remcnt);

  72         param->rw_offset = (uint64_t)off_low;
  73         param->rw_count = (uint32_t)count;
  74         param->rw_mincnt = 0;

  76         DTRACE_SMB_1(op__Read__start, smb_request_t *, sr); /* arg.rw */
  76         DTRACE_SMB_2(op__Read__start, smb_request_t *, sr,
  77             smb_rw_param_t *, param);

  78         return ((rc == 0) ? SDRC_SUCCESS : SDRC_ERROR);
  79 }

  81 void
  82 smb_post_read(smb_request_t *sr)
  83 {
  84         DTRACE_SMB_1(op__Read__done, smb_request_t *, sr); /* arg.rw */
  85         DTRACE_SMB_2(op__Read__done, smb_request_t *, sr,
  86             smb_rw_param_t *, sr->arg.rw);

  86         kmem_free(sr->arg.rw, sizeof (smb_rw_param_t));
  87 }
_____unchanged_portion_omitted_

 119 /*
 120  * Lock and read bytes from a file (SMB Core Plus).  The SmbLockAndRead/
 121  * SmbLockAndWrite sub-dialect is only valid on disk files: reject any
 122  * attempt to use it on non-disk shares.
 123  *
 124  * The requested count specifies the number of bytes desired.  Offset
 125  * specifies the offset in the file of the first byte to be locked then
 126  * read. Note that offset is limited to 32 bits, so this client request
 127  * is inappropriate for files with 64 bit offsets.
 128  *
 129  * As with SMB_LOCK_BYTE_RANGE request, if the lock cannot be granted
 130  * immediately an error should be returned to the client.  If an error
 131  * occurs on the lock, the bytes should not be read.
 132  *
 133  * On return, count is the number of bytes actually being returned, which
 134  * may be less than the count requested only if a read specifies bytes
 135  * beyond the current file size.  In this case only the bytes that exist
 136  * are returned.  A read completely beyond the end of file results in a
 137  * response of length zero.  This is the only circumstance when a zero
 138  * length response is generated.  A count returned which is less than the
 139  * count requested is the end of file indicator.
 140  */
 141 smb_sdrc_t
 142 smb_pre_lock_and_read(smb_request_t *sr)
```

```
143 {
144         smb_rw_param_t *param;
145         uint32_t off_low;
146         uint16_t count;
147         uint16_t remcnt;
148         int rc;

150         param = kmem_zalloc(sizeof (smb_rw_param_t), KM_SLEEP);
151         sr->arg.rw = param;

153         rc = smbsr_decode_vwv(sr, "wwlw", &sr->smb_fid,
154             &count, &off_low, &remcnt);

156         param->rw_offset = (uint64_t)off_low;
157         param->rw_count = (uint32_t)count;
158         param->rw_mincnt = 0;

160         DTRACE_SMB_1(op__LockAndRead__start, smb_request_t *, sr); /* arg.rw */
162         DTRACE_SMB_2(op__LockAndRead__start, smb_request_t *, sr,
163             smb_rw_param_t *, param);

162         return ((rc == 0) ? SDRC_SUCCESS : SDRC_ERROR);
163 }

165 void
166 smb_post_lock_and_read(smb_request_t *sr)
167 {
168         DTRACE_SMB_1(op__LockAndRead__done, smb_request_t *, sr); /* arg.rw */
171         DTRACE_SMB_2(op__LockAndRead__done, smb_request_t *, sr,
172             smb_rw_param_t *, sr->arg.rw);

170         kmem_free(sr->arg.rw, sizeof (smb_rw_param_t));
171 }
```

*unchanged_portion_omitted_*

```
221 /*
222  * The SMB_COM_READ_RAW protocol was a negotiated option introduced in
223  * SMB Core Plus to maximize performance when reading a large block
224  * of data from a server.  It's obsolete and no longer supported.
225  *
226  * We keep a handler for it so the dtrace provider can see if
227  * the client tried to use this command.
228  */
229 smb_sdrc_t
230 smb_pre_read_raw(smb_request_t *sr)
231 {
232         smb_rw_param_t *param;
233         uint32_t off_low;
234         uint32_t off_high;
235         uint32_t timeout;
236         uint16_t count;
237         int rc;

239         param = kmem_zalloc(sizeof (smb_rw_param_t), KM_SLEEP);
240         sr->arg.rw = param;
241         if (sr->smb_wct == 8) {
242                 rc = smbsr_decode_vwv(sr, "wlwwl2.", &sr->smb_fid,
243                     &off_low, &count, &param->rw_mincnt, &timeout);
244                 if (rc == 0) {
245                         param->rw_offset = (uint64_t)off_low;
246                         param->rw_count = (uint32_t)count;
247                 }
248         } else {
249                 rc = smbsr_decode_vwv(sr, "wlwwl2.l", &sr->smb_fid,
250                     &off_low, &count, &param->rw_mincnt, &timeout, &off_high);
251                 if (rc == 0) {
```



```
252                         param->rw_offset = ((uint64_t)off_high << 32) | off_low;
253                         param->rw_count = (uint32_t)count;
254                 }
255         }
256         DTRACE_SMB_1(op__ReadRaw__start, smb_request_t *, sr); /* arg.rw */
257         return (SDRC_SUCCESS);
258 }

260 void
261 smb_post_read_raw(smb_request_t *sr)
262 {
263         DTRACE_SMB_1(op__ReadRaw__done, smb_request_t *, sr); /* arg.rw */

265         kmem_free(sr->arg.rw, sizeof (smb_rw_param_t));
266 }

268 smb_sdrc_t
269 smb_com_read_raw(smb_request_t *sr)
270 {
271         smbsr_error(sr, NT_STATUS_NOT_SUPPORTED, ERRDOS,
272             ERROR_NOT_SUPPORTED);
273         return (SDRC_ERROR);
274 }

276 /*
277  * Read bytes from a file (SMB Core).  This request was extended in
278  * LM 0.12 to support 64-bit offsets, indicated by sending a wct of
279  * 12 and including additional offset information.
280  *
281  * MS-SMB 3.3.5.7 update to LM 0.12 4.2.4:
282  * If wct is 12 and CAP_LARGE_READX is set, the count may be larger
283  * than the negotiated buffer size.  If maxcnt_high is 0xFF, it must
284  * be ignored.  Otherwise, maxcnt_high represents the upper 16 bits
285  * of rw_count.
286  */
287 smb_sdrc_t
288 smb_pre_read_andx(smb_request_t *sr)
289 {
290         smb_rw_param_t *param;
291         uint32_t off_low;
292         uint32_t off_high;
293         uint32_t maxcnt_high;
294         uint16_t maxcnt_low;
295         uint16_t mincnt;
296         uint16_t remcnt;
297         int rc;

299         param = kmem_zalloc(sizeof (smb_rw_param_t), KM_SLEEP);
300         sr->arg.rw = param;

302         if (sr->smb_wct == 12) {
303                 rc = smbsr_decode_vwv(sr, "b3.wlwwlwl", &param->rw_andx,
304                     &sr->smb_fid, &off_low, &maxcnt_low, &mincnt, &maxcnt_high,
305                     &remcnt, &off_high);

307                 param->rw_offset = ((uint64_t)off_high << 32) |
308                     (uint64_t)off_low;

310                 param->rw_count = (uint32_t)maxcnt_low;

312                 if ((sr->session->capabilities & CAP_LARGE_READX) &&
313                     (maxcnt_high < 0xFF))
314                         param->rw_count |= maxcnt_high << 16;
315         } else {
316                 rc = smbsr_decode_vwv(sr, "b3.wlwwlw", &param->rw_andx,
317                     &sr->smb_fid, &off_low, &maxcnt_low, &mincnt, &maxcnt_high,
```

```
318                     &remcnt);

320                 param->rw_offset = (uint64_t)off_low;
321                 param->rw_count = (uint32_t)maxcnt_low;
322         }

324         param->rw_mincnt = 0;

326         DTRACE_SMB_1(op__ReadX__start, smb_request_t *, sr); /* arg.rw */
275         DTRACE_SMB_2(op__ReadX__start, smb_request_t *, sr,
276             smb_rw_param_t *, param);

328         return ((rc == 0) ? SDRC_SUCCESS : SDRC_ERROR);
329 }

331 void
332 smb_post_read_andx(smb_request_t *sr)
333 {
334         DTRACE_SMB_1(op__ReadX__done, smb_request_t *, sr); /* arg.rw */
284         DTRACE_SMB_2(op__ReadX__done, smb_request_t *, sr,
285             smb_rw_param_t *, sr->arg.rw);

336         kmem_free(sr->arg.rw, sizeof (smb_rw_param_t));
337 }
```

*unchanged_portion_omitted_*

```
**********************************************************
    6613 Fri Mar 31 14:29:43 2017
new/usr/src/uts/common/fs/smbsrv/smb_rename.c
Build provider 3rd arg from smb_request_t
hacking...
**********************************************************
```

```
/*
 * 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) 2007, 2010, Oracle and/or its affiliates. All rights reserved.
 * Copyright 2013 Nexenta Systems, Inc.  All rights reserved.
 */

#include <sys/synch.h>
#include <smbsrv/smb_kproto.h>
#include <smbsrv/smb_fsops.h>
#include <sys/nbmlock.h>

/*
 * NT_RENAME InformationLevels:
 *
 * SMB_NT_RENAME_MOVE_CLUSTER_INFO      Server returns invalid parameter.
 * SMB_NT_RENAME_SET_LINK_INFO          Create a hard link to a file.
 * SMB_NT_RENAME_RENAME_FILE            In-place rename of a file.
 * SMB_NT_RENAME_MOVE_FILE              Move (rename) a file.
 */
#define SMB_NT_RENAME_MOVE_CLUSTER_INFO 0x0102
#define SMB_NT_RENAME_SET_LINK_INFO     0x0103
#define SMB_NT_RENAME_RENAME_FILE       0x0104
#define SMB_NT_RENAME_MOVE_FILE         0x0105

/*
 * smb_com_rename
 *
 * Rename a file. Files OldFileName must exist and NewFileName must not.
 * Both pathnames must be relative to the Tid specified in the request.
 * Open files may be renamed.
 *
 * Multiple files may be renamed in response to a single request as Rename
 * File supports wildcards in the file name (last component of the path).
 * NOTE: we don’t support rename with wildcards.
 *
 * SearchAttributes indicates the attributes that the target file(s) must
 * have. If SearchAttributes is zero then only normal files are renamed.
 * If the system file or hidden attributes are specified then the rename
 * is inclusive - both the specified type(s) of files and normal files are
 * renamed.
 */
```



```
smb_sdrc_t
smb_pre_rename(smb_request_t *sr)
{
        smb_fqi_t *src_fqi = &sr->arg.dirop.fqi;
        smb_fqi_t *dst_fqi = &sr->arg.dirop.dst_fqi;
        int rc;

        if ((rc = smbsr_decode_vwv(sr, "w", &src_fqi->fq_sattr)) == 0) {
                rc = smbsr_decode_data(sr, "%SS", sr, &src_fqi->fq_path.pn_path,
                    &dst_fqi->fq_path.pn_path);

                dst_fqi->fq_sattr = 0;
        }

        DTRACE_SMB_1(op__Rename__start, smb_request_t *, sr); /* arg.dirop */
        DTRACE_SMB_2(op__Rename__start, smb_request_t *, sr,
            struct dirop *, &sr->arg.dirop);

        return ((rc == 0) ? SDRC_SUCCESS : SDRC_ERROR);
}
```

*_____unchanged_portion_omitted_*

```
/*
 * smb_com_nt_rename
 *
 * Rename a file. Files OldFileName must exist and NewFileName must not.
 * Both pathnames must be relative to the Tid specified in the request.
 * Open files may be renamed.
 *
 * SearchAttributes indicates the attributes that the target file(s) must
 * have. If SearchAttributes is zero then only normal files are renamed.
 * If the system file or hidden attributes are specified then the rename
 * is inclusive - both the specified type(s) of files and normal files are
 * renamed.
 */
smb_sdrc_t
smb_pre_nt_rename(smb_request_t *sr)
{
        smb_fqi_t *src_fqi = &sr->arg.dirop.fqi;
        smb_fqi_t *dst_fqi = &sr->arg.dirop.dst_fqi;
        uint32_t clusters;
        int rc;

        rc = smbsr_decode_vwv(sr, "wwl", &src_fqi->fq_sattr,
            &sr->arg.dirop.info_level, &clusters);
        if (rc == 0) {
                rc = smbsr_decode_data(sr, "%SS", sr,
                    &src_fqi->fq_path.pn_path, &dst_fqi->fq_path.pn_path);

                dst_fqi->fq_sattr = 0;
        }

        DTRACE_SMB_1(op__NtRename__start, smb_request_t *, sr); /* arg.dirop */
        DTRACE_SMB_2(op__NtRename__start, smb_request_t *, sr,
            struct dirop *, &sr->arg.dirop);

        return ((rc == 0) ? SDRC_SUCCESS : SDRC_ERROR);
}
```

*_____unchanged_portion_omitted_*

```
**********************************************************
    9031 Fri Mar 31 14:29:43 2017
new/usr/src/uts/common/fs/smbsrv/smb_session_setup_andx.c
NEX-1643 dtrace provider for smbsrv
Also illumos 1841:
 DTrace smb provider was mis-implemented, doesn't exist.
Add back handlers for read/write raw, so that
 legacy dtrace consumers can find the probes.
Kill extra arg in smb_negotiate
Fix missing "done" probe with smb_notify
Add example consumer: smb-trace.d
fix soi_pid
**********************************************************
/*
 * 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) 2007, 2010, Oracle and/or its affiliates. All rights reserved.
 * Copyright 2016 Nexenta Systems, Inc.  All rights reserved.
 */

#include <sys/types.h>
#include <sys/sid.h>
#include <sys/priv_names.h>
#include <sys/socket.h>
#include <netinet/in.h>
#include <smbsrv/smb_idmap.h>
#include <smbsrv/smb_kproto.h>
#include <smbsrv/smb_token.h>

smb_sdrc_t
smb_pre_session_setup_andx(smb_request_t *sr)
{
        smb_arg_sessionsetup_t  *sinfo;
        char                    *native_os;
        char                    *native_lm;
        int                     rc = 0;

        sinfo = smb_srm_zalloc(sr, sizeof (smb_arg_sessionsetup_t));
        sr->sr_ssetup = sinfo;

        /*
         * Enforce the minimum word count seen in the old protocol,
         * to make sure we have enough to decode the common stuff.
         * Further wcnt checks below.
         */
        if (sr->smb_wct < 10) {
                rc = -1;
                goto done;
        }
```



```
        }

        /*
         * Parse common part of SMB session setup.
         * skip: vcnumber(2), sesskey(4)
         */
        rc = smbsr_decode_vwv(sr, "b.www6.",
            &sr->andx_com, &sr->andx_off,
            &sinfo->ssi_maxbufsize, &sinfo->ssi_maxmpxcount);
        if (rc != 0)
                goto done;

        if (sr->session->dialect < NT_LM_0_12) {

                sinfo->ssi_type = SMB_SSNSETUP_PRE_NTLM012;
                sinfo->ssi_capabilities = 0;

                rc = smbsr_decode_vwv(sr, "w4.",
                    &sinfo->ssi_lmpwlen);
                if (rc != 0)
                        goto done;

                sinfo->ssi_lmpwd = smb_srm_zalloc(sr, sinfo->ssi_lmpwlen + 1);
                rc = smbsr_decode_data(sr, "%#c", sr, sinfo->ssi_lmpwlen,
                    sinfo->ssi_lmpwd);
                if (rc != 0)
                        goto done;

                sinfo->ssi_lmpwd[sinfo->ssi_lmpwlen] = 0;

                if (smbsr_decode_data(sr, "%u", sr, &sinfo->ssi_user) != 0)
                        sinfo->ssi_user = "";

                if (smbsr_decode_data(sr, "%u", sr, &sinfo->ssi_domain) != 0)
                        sinfo->ssi_domain = "";

                goto part2;
        }

        /*
         * We have dialect >= NT_LM_0_12
         */
        if (sr->smb_wct == 13) {
                /* Old style (non-extended) request. */
                sinfo->ssi_type = SMB_SSNSETUP_NTLM012_NOEXT;

                rc = smbsr_decode_vwv(sr, "ww4.l",
                    &sinfo->ssi_lmpwlen,
                    &sinfo->ssi_ntpwlen,
                    &sinfo->ssi_capabilities);
                if (rc != 0)
                        goto done;

                /* paranoid: ignore cap. ext. sec. here */
                sinfo->ssi_capabilities &= ~CAP_EXTENDED_SECURITY;

                sinfo->ssi_lmpwd = smb_srm_zalloc(sr, sinfo->ssi_lmpwlen + 1);
                sinfo->ssi_ntpwd = smb_srm_zalloc(sr, sinfo->ssi_ntpwlen + 1);

                rc = smbsr_decode_data(sr, "%#c#cuu", sr,
                    sinfo->ssi_lmpwlen, sinfo->ssi_lmpwd,
                    sinfo->ssi_ntpwlen, sinfo->ssi_ntpwd,
                    &sinfo->ssi_user, &sinfo->ssi_domain);
                if (rc != 0)
                        goto done;
```

```
120                 sinfo->ssi_lmpwd[sinfo->ssi_lmpwlen] = 0;
121                 sinfo->ssi_ntpwd[sinfo->ssi_ntpwlen] = 0;

123                 goto part2;
124         }

126         if (sr->smb_wct == 12) {
127                 /* New style (extended) request. */
128                 sinfo->ssi_type = SMB_SSNSETUP_NTLM012_EXTSEC;

130                 rc = smbsr_decode_vwv(sr, "w4.l",
131                     &sinfo->ssi_iseclen,
132                     &sinfo->ssi_capabilities);
133                 if (rc != 0)
134                         goto done;

136                 if ((sinfo->ssi_capabilities & CAP_EXTENDED_SECURITY) == 0) {
137                         rc = -1;
138                         goto done;
139                 }

141                 sinfo->ssi_isecblob = smb_srm_zalloc(sr, sinfo->ssi_iseclen);
142                 rc = smbsr_decode_data(sr, "%#c", sr,
143                     sinfo->ssi_iseclen, sinfo->ssi_isecblob);
144                 if (rc != 0)
145                         goto done;

147                 goto part2;
148         }

150         /* Invalid message */
151         rc = -1;
152         goto done;

154 part2:
155         /*
156          * Get the "Native OS" and "Native LanMan" strings.
157          * These are not critical to protocol function, so
158          * if we can't parse them, just guess "NT".
159          * These strings are free'd with the sr.
160          *
161          * In NTLM 0.12, the padding between the Native OS and Native LM
162          * is a bit strange.  On NT4.0, there is a 2 byte pad between the
163          * OS (Windows NT 1381) and LM (Windows NT 4.0).  On Windows 2000,
164          * there is no padding between the OS (Windows 2000 2195) and LM
165          * (Windows 2000 5.0). If the padding is removed from the decode
166          * string the NT4.0 LM comes out as an empty string.  So if the
167          * client's native OS is Win NT, assume extra padding.
168          */
169         rc = smbsr_decode_data(sr, "%u", sr, &native_os);
170         if (rc != 0 || native_os == NULL)
171                 sinfo->ssi_native_os = NATIVE_OS_WINNT;
172         else
173                 sinfo->ssi_native_os = smbnative_os_value(native_os);

175         if (sinfo->ssi_native_os == NATIVE_OS_WINNT)
176                 rc = smbsr_decode_data(sr, "%,u", sr, &native_lm);
177         else
178                 rc = smbsr_decode_data(sr, "%u", sr, &native_lm);
179         if (rc != 0 || native_lm == NULL)
180                 sinfo->ssi_native_lm = NATIVE_LM_NT;
181         else
182                 sinfo->ssi_native_lm = smbnative_lm_value(native_lm);
183         rc = 0;

185 done:
```



```
186         if (rc != 0) {
187                 cmn_err(CE_NOTE,
188                     "SmbSessonSetupX: client %s invalid request",
189                     sr->session->ip_addr_str);
190         }

192         DTRACE_SMB_1(op__SessionSetupX__start, smb_request_t *, sr);
192         DTRACE_SMB_2(op__SessionSetupX__start, smb_request_t *, sr,
193             smb_arg_sessionsetup_t, sinfo);
193         return ((rc == 0) ? SDRC_SUCCESS : SDRC_ERROR);
194 }

196 void
197 smb_post_session_setup_andx(smb_request_t *sr)
198 {
199         smb_arg_sessionsetup_t  *sinfo = sr->sr_ssetup;

201         DTRACE_SMB_1(op__SessionSetupX__done, smb_request_t *, sr);
202         DTRACE_SMB_2(op__SessionSetupX__done, smb_request_t *, sr,
203             smb_arg_sessionsetup_t, sinfo);

203         if (sinfo->ssi_lmpwd != NULL)
204                 bzero(sinfo->ssi_lmpwd, sinfo->ssi_lmpwlen);

206         if (sinfo->ssi_ntpwd != NULL)
207                 bzero(sinfo->ssi_ntpwd, sinfo->ssi_ntpwlen);
208 }
```

*unchanged_portion_omitted_*

```
**********************************************************
   15508 Fri Mar 31 14:29:43 2017
new/usr/src/uts/common/fs/smbsrv/smb_tree_connect.c
NEX-1643 dtrace provider for smbsrv
Also illumos 1841:
 DTrace smb provider was mis-implemented, doesn't exist.
Add back handlers for read/write raw, so that
 legacy dtrace consumers can find the probes.
Kill extra arg in smb_negotiate
Fix missing "done" probe with smb_notify
Add example consumer: smb-trace.d
fix soi_pid
**********************************************************
______unchanged_portion_omitted_
```

```
  56 /*
  57  * SmbTreeConnect: Map a share to a tree and obtain a tree-id (TID).
  58  *
  59  * Client Request                     Description
  60  * ================================== ==================================
  61  *
  62  * UCHAR WordCount;                   Count of parameter words = 0
  63  * USHORT ByteCount;                  Count of data bytes;    min = 4
  64  * UCHAR BufferFormat1;               0x04
  65  * STRING Path[];                     Server name and share name
  66  * UCHAR BufferFormat2;               0x04
  67  * STRING Password[];                 Password
  68  * UCHAR BufferFormat3;               0x04
  69  * STRING Service[];                  Service name
  70  *
  71  * The CIFS server responds with:
  72  *
  73  * Server Response                  Description
  74  * ================================ ================================
  75  *
  76  * UCHAR WordCount;                 Count of parameter words = 2
  77  * USHORT MaxBufferSize;            Max size message the server handles
  78  * USHORT Tid;                      Tree ID
  79  * USHORT ByteCount;                Count of data bytes = 0
  80  *
  81  * If the negotiated dialect is MICROSOFT NETWORKS 1.03 or earlier,
  82  * MaxBufferSize in the response message indicates the maximum size
  83  * message that the server can handle.  The client should not generate
  84  * messages, nor expect to receive responses, larger than this.  This
  85  * must be constant for a given server. For newer dialects, this field
  86  * is ignored.
  87  */
  88 smb_sdrc_t
  89 smb_pre_tree_connect(smb_request_t *sr)
  90 {
  91         smb_arg_tcon_t  *tcon = &sr->sr_tcon;
  92         int             rc;

  94         /*
  95          * Perhaps this should be "%A.sA" now that unicode is enabled.
  96          */
  97         rc = smbsr_decode_data(sr, "%AAA", sr, &tcon->path,
  98             &tcon->password, &tcon->service);

 100         tcon->flags = 0;
 101         tcon->optional_support = 0;

 103         DTRACE_SMB_1(op__TreeConnect__start, smb_request_t *, sr);
 103         DTRACE_SMB_2(op__TreeConnect__start, smb_request_t *, sr,
 104             smb_arg_tcon_t *, tcon);
```



```
 105         return ((rc == 0) ? SDRC_SUCCESS : SDRC_ERROR);
 106 }
______unchanged_portion_omitted_
```

```
 135 /*
 136  * SmbTreeConnectX: Map a share to a tree and obtain a tree-id (TID).
 137  *
 138  * Client Request                     Description
 139  * ================================== ==================================
 140  *
 141  * UCHAR WordCount;                   Count of parameter words = 4
 142  * UCHAR AndXCommand;                 Secondary (X) command; 0xFF = none
 143  * UCHAR AndXReserved;                Reserved (must be 0)
 144  * USHORT AndXOffset;                 Offset to next command WordCount
 145  * USHORT Flags;                      Additional information
 146  *                                    bit 0 set = disconnect Tid
 147  * USHORT PasswordLength;             Length of Password[]
 148  * USHORT ByteCount;                  Count of data bytes;    min = 3
 149  * UCHAR Password[];                  Password
 150  * STRING Path[];                     Server name and share name
 151  * STRING Service[];                  Service name
 152  *
 153  * If the negotiated dialect is LANMAN1.0 or later, then it is a protocol
 154  * violation for the client to send this message prior to a successful
 155  * SMB_COM_SESSION_SETUP_ANDX, and the server ignores Password.
 156  *
 157  * If the negotiated dialect is prior to LANMAN1.0 and the client has not
 158  * sent a successful SMB_COM_SESSION_SETUP_ANDX request when the tree
 159  * connect arrives, a user level security mode server must nevertheless
 160  * validate the client's credentials.
 161  *
 162  * Flags (prefix with TREE_CONNECT_ANDX_):
 163  * ========================  ========================================
 164  * 0x0001 DISCONECT_TID      The tree specified by TID in the SMB header
 165  *                           should be disconnected - disconnect errors
 166  *                           should be ignored.
 167  *
 168  * 0x0004 EXTENDED_SIGNATURES  Client request for signing key protection.
 169  *
 170  * 0x0008 EXTENDED_RESPONSE    Client request for extended information.
 171  *
 172  * Path follows UNC style syntax (\\server\share) and indicates the name
 173  * of the resource to which the client wishes to connect.
 174  *
 175  * Because Password may be an authentication response, it is a variable
 176  * length field with the length specified by PasswordLength.   If
 177  * authentication is not being used, Password should be a null terminated
 178  * ASCII string with PasswordLength set to the string size including the
 179  * terminating null.
 180  *
 181  * The server can enforce whatever policy it desires to govern share
 182  * access.  Administrative privilege is required for administrative
 183  * shares (C$, etc.).
 184  *
 185  * The Service component indicates the type of resource the client
 186  * intends to access.  Valid values are:
 187  *
 188  * Service   Description              Earliest Dialect Allowed
 189  * ========  =======================  ================================
 190  *
 191  * A:        disk share               PC NETWORK PROGRAM 1.0
 192  * LPT1:     printer                  PC NETWORK PROGRAM 1.0
 193  * IPC       named pipe               MICROSOFT NETWORKS 3.0
 194  * COMM      communications device    MICROSOFT NETWORKS 3.0
 195  * ?????     any type of device       MICROSOFT NETWORKS 3.0
 196  *
```

```
197  * If the negotiated dialect is earlier than DOS LANMAN2.1, the response to
198  * this SMB is:
199  *
200  * Server Response                  Description
201  * ================================ =================================
202  *
203  * UCHAR WordCount;                 Count of parameter words = 2
204  * UCHAR AndXCommand;               Secondary (X) command;  0xFF = none
205  * UCHAR AndXReserved;              Reserved (must be 0)
206  * USHORT AndXOffset;               Offset to next command WordCount
207  * USHORT ByteCount;                Count of data bytes;    min = 3
208  *
209  * If the negotiated is DOS LANMAN2.1 or later, the response to this SMB
210  * is:
211  *
212  * Server Response                  Description
213  * ================================ =================================
214  *
215  * UCHAR WordCount;                 Count of parameter words = 3
216  * UCHAR AndXCommand;               Secondary (X) command;  0xFF = none
217  * UCHAR AndXReserved;              Reserved (must be 0)
218  * USHORT AndXOffset;               Offset to next command WordCount
219  * USHORT OptionalSupport;          Optional support bits
220  * USHORT ByteCount;                Count of data bytes;    min = 3
221  * UCHAR Service[];                 Service type connected to.  Always
222  *                                   ANSII.
223  * STRING NativeFileSystem[];       Native file system for this tree
224  *
225  * NativeFileSystem is the name of the filesystem; values to be expected
226  * include FAT, NTFS, etc.
227  *
228  * OptionalSupport:
229  * ============================  =========================
230  * 0x0001 SMB_SUPPORT_SEARCH_BITS  The server supports the use of Search
231  *                                 Attributes in client requests.
232  * 0x0002 SMB_SHARE_IS_IN_DFS      The share is managed by DFS.
233  * 0x000C SMB_CSC_MASK             Offline-caching mask - see CSC flags.
234  * 0x0010 SMB_UNIQUE_FILE_NAME     The server uses long names and does not
235  *                                 support short names.  Indicator for
236  *                                 clients directory/name-space caching.
237  * 0x0020 SMB_EXTENDED_SIGNATURES  The server will use signing key protection.
238  *
239  * Client-side caching (offline files):
240  * ============================  =========================
241  * 0x0000 SMB_CSC_CACHE_MANUAL_REINT Clients may cache files for offline use
242  *                                 but automatic file-by-file reintegration
243  *                                 is not allowed.
244  * 0x0004 SMB_CSC_CACHE_AUTO_REINT Automatic file-by-file reintegration is
245  *                                 allowed.
246  * 0x0008 SMB_CSC_CACHE_VDO        File opens do not need to be flowed.
247  * 0x000C SMB_CSC_CACHE_NONE       CSC is disabled for this share.
248  *
249  * Some servers negotiate "DOS LANMAN2.1" dialect or later and still send
250  * the "downlevel" (i.e. wordcount==2) response.  Valid AndX following
251  * commands are
252  *
253  * SMB_COM_OPEN              SMB_COM_OPEN_ANDX          SMB_COM_CREATE
254  * SMB_COM_CREATE_NEW        SMB_COM_CREATE_DIRECTORY   SMB_COM_DELETE
255  * SMB_COM_DELETE_DIRECTORY  SMB_COM_FIND               SMB_COM_COPY
256  * SMB_COM_FIND_UNIQUE       SMB_COM_RENAME
257  * SMB_COM_CHECK_DIRECTORY   SMB_COM_QUERY_INFORMATION
258  * SMB_COM_GET_PRINT_QUEUE   SMB_COM_OPEN_PRINT_FILE
259  * SMB_COM_TRANSACTION       SMB_COM_NO_ANDX_CMD
260  * SMB_COM_SET_INFORMATION   SMB_COM_NT_RENAME
261  *
262  * Errors:
```



```
263  * ERRDOS/ERRnomem
264  * ERRDOS/ERRbadpath
265  * ERRDOS/ERRinvdevice
266  * ERRSRV/ERRaccess
267  * ERRSRV/ERRbadpw
268  * ERRSRV/ERRinvnetname
269  */
270 smb_sdrc_t
271 smb_pre_tree_connect_andx(smb_request_t *sr)
272 {
273         smb_arg_tcon_t  *tcon = &sr->sr_tcon;
274         uint8_t         *pwbuf = NULL;
275         uint16_t        pwlen = 0;
276         int             rc;

278         rc = smbsr_decode_vwv(sr, "b.www", &sr->andx_com, &sr->andx_off,
279             &tcon->flags, &pwlen);
280         if (rc == 0) {
281                 if (pwlen != 0)
282                         pwbuf = smb_srm_zalloc(sr, pwlen);

284                 rc = smbsr_decode_data(sr, "%#cus", sr, pwlen, pwbuf,
285                     &tcon->path, &tcon->service);

287                 tcon->pwdlen = pwlen;
288                 tcon->password = (char *)pwbuf;
289         }

291         tcon->optional_support = 0;

293         DTRACE_SMB_1(op__TreeConnectX__start, smb_request_t *, sr);
294         DTRACE_SMB_2(op__TreeConnectX__start, smb_request_t *, sr,
295             smb_arg_tcon_t *, tcon);

295         return ((rc == 0) ? SDRC_SUCCESS : SDRC_ERROR);
296 }
```

*_____unchanged_portion_omitted_*

```
**********************************************************
   17019 Fri Mar 31 14:29:44 2017
new/usr/src/uts/common/fs/smbsrv/smb_write.c
Build provider 3rd arg from smb_request_t
hacking...
NEX-1643 dtrace provider for smbsrv
Also illumos 1841:
 DTrace smb provider was mis-implemented, doesn't exist.
Add back handlers for read/write raw, so that
 legacy dtrace consumers can find the probes.
Kill extra arg in smb_negotiate
Fix missing "done" probe with smb_notify
Add example consumer: smb-trace.d
fix soi_pid
**********************************************************
```

```
   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) 2007, 2010, Oracle and/or its affiliates. All rights reserved.
  24  * Copyright 2016 Nexenta Systems, Inc.  All rights reserved.
  25  */

  27 #include <sys/sdt.h>
  28 #include <smbsrv/smb_kproto.h>
  29 #include <smbsrv/smb_fsops.h>
  30 #include <smbsrv/netbios.h>


  33 static int smb_write_truncate(smb_request_t *, smb_rw_param_t *);


  36 /*
  37  * Write count bytes at the specified offset in a file.  The offset is
  38  * limited to 32-bits.  If the count is zero, the file is truncated to
  39  * the length specified by the offset.
  40  *
  41  * The response count indicates the actual number of bytes written, which
  42  * will equal the requested count on success.  If request and response
  43  * counts differ but there is no error, the client will assume that the
  44  * server encountered a resource issue.
  45  */
  46 smb_sdrc_t
  47 smb_pre_write(smb_request_t *sr)
  48 {
  49         smb_rw_param_t *param;
  50         uint32_t off;
  51         uint16_t count;
```



```
  52         int rc;

  54         param = kmem_zalloc(sizeof (smb_rw_param_t), KM_SLEEP);
  55         sr->arg.rw = param;
  56         param->rw_magic = SMB_RW_MAGIC;

  58         rc = smbsr_decode_vwv(sr, "wwl", &sr->smb_fid, &count, &off);

  60         param->rw_count = (uint32_t)count;
  61         param->rw_offset = (uint64_t)off;
  62         param->rw_vdb.vdb_uio.uio_loffset = (offset_t)param->rw_offset;

  64         DTRACE_SMB_1(op__Write__start, smb_request_t *, sr); /* arg.rw */
  64         DTRACE_SMB_2(op__Write__start, smb_request_t *, sr,
  65             smb_rw_param_t *, param);

  66         return ((rc == 0) ? SDRC_SUCCESS : SDRC_ERROR);
  67 }

  69 void
  70 smb_post_write(smb_request_t *sr)
  71 {
  72         DTRACE_SMB_1(op__Write__done, smb_request_t *, sr); /* arg.rw */
  73         DTRACE_SMB_2(op__Write__done, smb_request_t *, sr,
  74             smb_rw_param_t *, sr->arg.rw);

  74         kmem_free(sr->arg.rw, sizeof (smb_rw_param_t));
  75 }
_____unchanged_portion_omitted_

 118 /*
 119  * Write count bytes to a file and then close the file.  This function
 120  * can only be used to write to 32-bit offsets and the client must set
 121  * WordCount (6 or 12) correctly in order to locate the data to be
 122  * written.  If an error occurs on the write, the file should still be
 123  * closed.  If Count is 0, the file is truncated (or extended) to offset.
 124  *
 125  * If the last_write time is non-zero, last_write should be used to set
 126  * the mtime.  Otherwise the file system stamps the mtime.  Failure to
 127  * set mtime should not result in an error response.
 128  */
 129 smb_sdrc_t
 130 smb_pre_write_and_close(smb_request_t *sr)
 131 {
 132         smb_rw_param_t *param;
 133         uint32_t off;
 134         uint16_t count;
 135         int rc;

 137         param = kmem_zalloc(sizeof (smb_rw_param_t), KM_SLEEP);
 138         sr->arg.rw = param;
 139         param->rw_magic = SMB_RW_MAGIC;

 141         if (sr->smb_wct == 12) {
 142                 rc = smbsr_decode_vwv(sr, "wwll12.", &sr->smb_fid,
 143                     &count, &off, &param->rw_last_write);
 144         } else {
 145                 rc = smbsr_decode_vwv(sr, "wwll", &sr->smb_fid,
 146                     &count, &off, &param->rw_last_write);
 147         }

 149         param->rw_count = (uint32_t)count;
 150         param->rw_offset = (uint64_t)off;

 152         DTRACE_SMB_1(op__WriteAndClose__start, smb_request_t *, sr); /* arg.rw *
 154         DTRACE_SMB_2(op__WriteAndClose__start, smb_request_t *, sr,
```

```
155             smb_rw_param_t *, param);

154         return ((rc == 0) ? SDRC_SUCCESS : SDRC_ERROR);
155 }

157 void
158 smb_post_write_and_close(smb_request_t *sr)
159 {
160         DTRACE_SMB_1(op__WriteAndClose__done, smb_request_t *, sr); /* arg.rw */
163         DTRACE_SMB_2(op__WriteAndClose__done, smb_request_t *, sr,
164             smb_rw_param_t *, sr->arg.rw);

162         kmem_free(sr->arg.rw, sizeof (smb_rw_param_t));
163 }
```

*unchanged_portion_omitted_*

```
213 /*
214  * Write count bytes to a file at the specified offset and then unlock
215  * them.  Write behind is safe because the client should have the range
216  * locked and this request is allowed to extend the file - note that
217  * offset is limited to 32-bits.
218  *
219  * Spec advice: it is an error for count to be zero.  For compatibility,
220  * we take no action and return success.
221  *
222  * The SmbLockAndRead/SmbWriteAndUnlock sub-dialect is only valid on disk
223  * files.  Reject any attempt to use it on other shares.
224  *
225  * The response count indicates the actual number of bytes written, which
226  * will equal the requested count on success.  If request and response
227  * counts differ but there is no error, the client will assume that the
228  * server encountered a resource issue.
229  */
230 smb_sdrc_t
231 smb_pre_write_and_unlock(smb_request_t *sr)
232 {
233         smb_rw_param_t *param;
234         uint32_t off;
235         uint16_t count;
236         uint16_t remcnt;
237         int rc;

239         param = kmem_zalloc(sizeof (smb_rw_param_t), KM_SLEEP);
240         sr->arg.rw = param;
241         param->rw_magic = SMB_RW_MAGIC;

243         rc = smbsr_decode_vwv(sr, "wwlw", &sr->smb_fid, &count, &off, &remcnt);

245         param->rw_count = (uint32_t)count;
246         param->rw_offset = (uint64_t)off;

248         DTRACE_SMB_1(op__WriteAndUnlock__start, smb_request_t *, sr); /* arg.rw
252         DTRACE_SMB_2(op__WriteAndUnlock__start, smb_request_t *, sr,
253             smb_rw_param_t *, param);

250         return ((rc == 0) ? SDRC_SUCCESS : SDRC_ERROR);
251 }

253 void
254 smb_post_write_and_unlock(smb_request_t *sr)
255 {
256         DTRACE_SMB_1(op__WriteAndUnlock__done, smb_request_t *, sr); /* arg.rw *
261         DTRACE_SMB_2(op__WriteAndUnlock__done, smb_request_t *, sr,
262             smb_rw_param_t *, sr->arg.rw);

258         kmem_free(sr->arg.rw, sizeof (smb_rw_param_t));
```



```
259 }
```

*unchanged_portion_omitted_*

```
321 /*
322  * The SMB_COM_WRITE_RAW protocol was a negotiated option introduced in
323  * SMB Core Plus to maximize performance when writing a large block
324  * of data to a server.  It's obsolete and no longer supported.
325  *
326  * We keep a handler for it so the dtrace provider can see if
327  * the client tried to use this command.
328  */
329 smb_sdrc_t
330 smb_pre_write_raw(smb_request_t *sr)
331 {
332         smb_rw_param_t *param;
333         uint32_t off_low;
334         uint32_t timeout;
335         uint32_t off_high;
336         uint16_t datalen;
337         uint16_t total;
338         int rc;

340         param = kmem_zalloc(sizeof (smb_rw_param_t), KM_SLEEP);
341         sr->arg.rw = param;

343         if (sr->smb_wct == 12) {
344                 rc = smbsr_decode_vwv(sr, "ww2.llw4.ww", &sr->smb_fid, &total,
345                     &off_low, &timeout, &param->rw_mode, &datalen,
346                     &param->rw_dsoff);

348                 param->rw_offset = (uint64_t)off_low;
349         } else {
350                 rc = smbsr_decode_vwv(sr, "ww2.llw4.wwl", &sr->smb_fid, &total,
351                     &off_low, &timeout, &param->rw_mode, &datalen,
352                     &param->rw_dsoff, &off_high);

354                 param->rw_offset = ((uint64_t)off_high << 32) | off_low;
355         }

357         param->rw_count = (uint32_t)datalen;
358         param->rw_total = (uint32_t)total;

360         DTRACE_SMB_1(op__WriteRaw__start, smb_request_t *, sr); /* arg.rw */

362         return ((rc == 0) ? SDRC_SUCCESS : SDRC_ERROR);
363 }

365 void
366 smb_post_write_raw(smb_request_t *sr)
367 {
368         DTRACE_SMB_1(op__WriteRaw__done, smb_request_t *, sr); /* arg.rw */

370         kmem_free(sr->arg.rw, sizeof (smb_rw_param_t));
371 }

373 smb_sdrc_t
374 smb_com_write_raw(struct smb_request *sr)
375 {
376         smbsr_error(sr, NT_STATUS_NOT_SUPPORTED, ERRDOS,
377             ERROR_NOT_SUPPORTED);
378         return (SDRC_ERROR);
379 }

381 /*
382  * Write bytes to a file (SMB Core).  This request was extended in
383  * LM 0.12 to support 64-bit offsets, indicated by sending a wct of
```

```
 384  * 14, instead of 12, and including additional offset information.
 385  *
 386  * A ByteCount of 0 does not truncate the file - use SMB_COM_WRITE
 387  * to truncate a file.  A zero length merely transfers zero bytes.
 388  *
 389  * If bit 0 of WriteMode is set, Fid must refer to a disk file and
 390  * the data must be on stable storage before responding.
 391  *
 392  * MS-SMB 3.3.5.8 update to LM 0.12 4.2.5:
 393  * If CAP_LARGE_WRITEX is set, the byte count may be larger than the
 394  * negotiated buffer size and the server is expected to write the
 395  * number of bytes specified.
 396  */
 397 smb_sdrc_t
 398 smb_pre_write_andx(smb_request_t *sr)
 399 {
 400         smb_rw_param_t *param;
 401         uint32_t off_low;
 402         uint32_t off_high;
 403         uint16_t datalen_low;
 404         uint16_t datalen_high;
 405         uint16_t remcnt;
 406         int rc;

 408         param = kmem_zalloc(sizeof (smb_rw_param_t), KM_SLEEP);
 409         sr->arg.rw = param;
 410         param->rw_magic = SMB_RW_MAGIC;

 412         if (sr->smb_wct == 14) {
 413                 rc = smbsr_decode_vwv(sr, "4.wl4.wwwwwl", &sr->smb_fid,
 414                     &off_low, &param->rw_mode, &remcnt, &datalen_high,
 415                     &datalen_low, &param->rw_dsoff, &off_high);

 417                 if (param->rw_dsoff >= 63)
 418                         param->rw_dsoff -= 63;
 419                 param->rw_offset = ((uint64_t)off_high << 32) | off_low;
 420         } else if (sr->smb_wct == 12) {
 421                 rc = smbsr_decode_vwv(sr, "4.wl4.wwwww", &sr->smb_fid,
 422                     &off_low, &param->rw_mode, &remcnt, &datalen_high,
 423                     &datalen_low, &param->rw_dsoff);

 425                 if (param->rw_dsoff >= 59)
 426                         param->rw_dsoff -= 59;
 427                 param->rw_offset = (uint64_t)off_low;
 428                 /* off_high not present */
 429         } else {
 430                 rc = -1;
 431         }

 433         param->rw_count = (uint32_t)datalen_low;

 435         /*
 436          * Work-around a Win7 bug, where it fails to set the
 437          * CAP_LARGE_WRITEX flag during session setup.  Assume
 438          * a large write if the data remaining is >= 64k.
 439          */
 440         if ((sr->session->capabilities & CAP_LARGE_WRITEX) != 0 ||
 441             (sr->smb_data.max_bytes > (sr->smb_data.chain_offset + 0xFFFF)))
 442                 param->rw_count |= ((uint32_t)datalen_high << 16);

 444         DTRACE_SMB_1(op__WriteX__start, smb_request_t *, sr); /* arg.rw */
 390         DTRACE_SMB_2(op__WriteX__start, smb_request_t *, sr,
 391             smb_rw_param_t *, param);

 446         return ((rc == 0) ? SDRC_SUCCESS : SDRC_ERROR);
 447 }
```



```
 449 void
 450 smb_post_write_andx(smb_request_t *sr)
 451 {
 452         DTRACE_SMB_1(op__WriteX__done, smb_request_t *, sr); /* arg.rw */
 399         DTRACE_SMB_2(op__WriteX__done, smb_request_t *, sr,
 400             smb_rw_param_t *, sr->arg.rw);

 454         kmem_free(sr->arg.rw, sizeof (smb_rw_param_t));
 455 }
______unchanged_portion_omitted_
```

```
**********************************************************
   37129 Fri Mar 31 14:29:44 2017
new/usr/src/uts/common/smbsrv/smb_kproto.h
NEX-1643 dtrace provider for smbsrv
Also illumos 1841:
 DTrace smb provider was mis-implemented, doesn't exist.
Add back handlers for read/write raw, so that
 legacy dtrace consumers can find the probes.
Kill extra arg in smb_negotiate
Fix missing "done" probe with smb_notify
Add example consumer: smb-trace.d
fix soi_pid
**********************************************************
/*
 * 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) 2007, 2010, Oracle and/or its affiliates. All rights reserved.
 * Copyright 2016 Syneto S.R.L.  All rights reserved.
 * Copyright 2017 Nexenta Systems, Inc.  All rights reserved.
 */

/*
 * Function prototypes for the SMB module.
 */

#ifndef _SMB_KPROTO_H_
#define _SMB_KPROTO_H_

#ifdef __cplusplus
extern "C" {
#endif

#include <sys/types.h>
#include <sys/param.h>
#include <sys/systm.h>
#include <sys/debug.h>
#include <sys/kmem.h>
#include <sys/socket.h>
#include <sys/ksocket.h>
#include <sys/cred.h>
#include <sys/nbmlock.h>
#include <sys/sunddi.h>
#include <sys/atomic.h>
#include <smbsrv/smb.h>
#include <smbsrv/string.h>
#include <smbsrv/smb_vops.h>
#include <smbsrv/smb_xdr.h>
```



```
#include <smbsrv/smb_token.h>
#include <smbsrv/smb_ktypes.h>
#include <smbsrv/smb_ioctl.h>

/*
 * DTrace SDT probes have different signatures in userland than they do in
 * kernel.  If we're compiling for user mode (libfksmbsrv) define them as
 * either no-op (for the SMB dtrace provider) or libfksmbsrv functions for
 * the other SDT probe sites.
 */
#ifndef _KERNEL

extern void smb_dtrace1(const char *f, const char *n,
                        const char *t1, long v1);
extern void smb_dtrace2(const char *f, const char *n,
                        const char *t1, long v1,
                        const char *t2, long v2);
extern void smb_dtrace3(const char *f, const char *n,
                        const char *t1, long v1,
                        const char *t2, long v2,
                        const char *t3, long v3);

/*
 * These are for the SMB dtrace proivder, which for a user-mode build
 * are largely redundant with the fbt probes so make these no-ops.
 */
#undef  DTRACE_SMB_1
#define DTRACE_SMB_1(n, a, b)                   ((void)b)
#undef  DTRACE_SMB_2
#define DTRACE_SMB_2(n, a, b, c, d)             ((void)b, (void)d)

/*
 * These are for the other (specialized) dtrace SDT probes sprinkled
 * through the smbsrv code.  In libfksmbsrv map these to functions.
 */

#undef  DTRACE_PROBE1
#define DTRACE_PROBE1(n, a, b) \
        smb_dtrace1(__func__, #n, #a, (long)b)

#undef  DTRACE_PROBE2
#define DTRACE_PROBE2(n, a, b, c, d) \
        smb_dtrace2(__func__, #n, #a, (long)b, #c, (long)d)

#undef  DTRACE_PROBE3
#define DTRACE_PROBE3(n, a, b, c, d, e, f) \
        smb_dtrace3(__func__, #n, #a, (long)b, #c, (long)d, #e, (long)f)

#endif  /* _KERNEL */

extern  int smb_maxbufsize;
extern  int smb_flush_required;
extern  int smb_dirsymlink_enable;
extern  int smb_oplock_levelII;
extern  int smb_oplock_timeout;
extern  int smb_oplock_min_timeout;
extern  int smb_shortnames;
extern  int smb_sign_debug;
extern  int smb_raw_mode;
extern  uint_t smb_audit_flags;
extern  int smb_ssetup_threshold;
extern  int smb_tcon_threshold;
extern  int smb_opipe_threshold;
extern  int smb_ssetup_timeout;
extern  int smb_tcon_timeout;
extern  int smb_opipe_timeout;
```

```
119 extern const uint32_t smb_vop_dosattr_settable;

121 /* Thread priorities - see smb_init.c */
122 extern  int smbsrv_base_pri;
123 extern  int smbsrv_listen_pri;
124 extern  int smbsrv_receive_pri;
125 extern  int smbsrv_worker_pri;
126 extern  int smbsrv_notify_pri;
127 extern  int smbsrv_timer_pri;

129 extern  kmem_cache_t            *smb_cache_request;
130 extern  kmem_cache_t            *smb_cache_session;
131 extern  kmem_cache_t            *smb_cache_user;
132 extern  kmem_cache_t            *smb_cache_tree;
133 extern  kmem_cache_t            *smb_cache_ofile;
134 extern  kmem_cache_t            *smb_cache_odir;
135 extern  kmem_cache_t            *smb_cache_opipe;
136 extern  kmem_cache_t            *smb_cache_event;
137 extern  kmem_cache_t            *smb_cache_lock;

139 extern  kmem_cache_t            *smb_kshare_cache_vfs;

141 time_t smb_get_boottime(void);
142 int smb_server_lookup(smb_server_t **);
143 void smb_server_release(smb_server_t *);

145 /*
146  * SMB request handers called from the dispatcher.  Each SMB request
147  * is handled in three phases: pre, com (command) and post.
148  *
149  * The pre-handler is primarily to set things up for the DTrace start
150  * probe.  Typically, the SMB request is unmarshalled so that request
151  * specific context can be traced.  This is also a useful place to
152  * allocate memory that will be used throughout the processing of the
153  * command.
154  *
155  * The com-handler performs the requested operation: request validation,
156  * bulk (write) incoming data decode, implementation of the appropriate
157  * algorithm and transmission of a response (as appropriate).
158  *
159  * The post-handler is always called, regardless of success or failure
160  * of the pre or com functions, to trigger the DTrace done probe and
161  * deallocate memory allocated in the pre-handler.
162  */
163 #define SMB_SDT_OPS(NAME)       \
164         smb_pre_##NAME,         \
165         smb_com_##NAME,         \
166         smb_post_##NAME

168 #define SMB_COM_DECL(NAME)                              \
169         smb_sdrc_t smb_pre_##NAME(smb_request_t *);     \
170         smb_sdrc_t smb_com_##NAME(smb_request_t *);     \
171         void smb_post_##NAME(smb_request_t *)

173 SMB_COM_DECL(check_directory);
174 SMB_COM_DECL(close);
175 SMB_COM_DECL(close_and_tree_disconnect);
176 SMB_COM_DECL(close_print_file);
177 SMB_COM_DECL(create);
178 SMB_COM_DECL(create_directory);
179 SMB_COM_DECL(create_new);
180 SMB_COM_DECL(create_temporary);
181 SMB_COM_DECL(delete);
182 SMB_COM_DECL(delete_directory);
183 SMB_COM_DECL(echo);
184 SMB_COM_DECL(find);
```



```
185 SMB_COM_DECL(find_close);
186 SMB_COM_DECL(find_close2);
187 SMB_COM_DECL(find_unique);
188 SMB_COM_DECL(flush);
189 SMB_COM_DECL(get_print_queue);
190 SMB_COM_DECL(invalid);
191 SMB_COM_DECL(ioctl);
192 SMB_COM_DECL(lock_and_read);
193 SMB_COM_DECL(lock_byte_range);
194 SMB_COM_DECL(locking_andx);
195 SMB_COM_DECL(logoff_andx);
196 SMB_COM_DECL(negotiate);
197 SMB_COM_DECL(nt_cancel);
198 SMB_COM_DECL(nt_create_andx);
199 SMB_COM_DECL(nt_rename);
200 SMB_COM_DECL(nt_transact);
201 SMB_COM_DECL(nt_transact_secondary);
202 SMB_COM_DECL(open);
203 SMB_COM_DECL(open_andx);
204 SMB_COM_DECL(open_print_file);
205 SMB_COM_DECL(process_exit);
206 SMB_COM_DECL(query_information);
207 SMB_COM_DECL(query_information2);
208 SMB_COM_DECL(query_information_disk);
209 SMB_COM_DECL(read);
210 SMB_COM_DECL(read_andx);
211 SMB_COM_DECL(read_raw);
212 SMB_COM_DECL(rename);
213 SMB_COM_DECL(search);
214 SMB_COM_DECL(seek);
215 SMB_COM_DECL(session_setup_andx);
216 SMB_COM_DECL(set_information);
217 SMB_COM_DECL(set_information2);
218 SMB_COM_DECL(transaction);
219 SMB_COM_DECL(transaction2);
220 SMB_COM_DECL(transaction2_secondary);
221 SMB_COM_DECL(transaction_secondary);
222 SMB_COM_DECL(tree_connect);
223 SMB_COM_DECL(tree_connect_andx);
224 SMB_COM_DECL(tree_disconnect);
225 SMB_COM_DECL(unlock_byte_range);
226 SMB_COM_DECL(write);
227 SMB_COM_DECL(write_and_close);
228 SMB_COM_DECL(write_and_unlock);
229 SMB_COM_DECL(write_andx);
230 SMB_COM_DECL(write_print_file);
231 SMB_COM_DECL(write_raw);

233 #define SMB_NT_TRANSACT_DECL(NAME)                                      \
234         smb_sdrc_t smb_pre_##NAME(smb_request_t *, smb_xa_t *); \
235         smb_sdrc_t smb_##NAME(smb_request_t *, smb_xa_t *);     \
236         void smb_post_##NAME(smb_request_t *, smb_xa_t *)

238 SMB_NT_TRANSACT_DECL(nt_transact_create);

240 smb_sdrc_t smb_nt_transact_notify_change(smb_request_t *, smb_xa_t *);
241 smb_sdrc_t smb_nt_transact_query_security_info(smb_request_t *, smb_xa_t *);
242 smb_sdrc_t smb_nt_transact_set_security_info(smb_request_t *, smb_xa_t *);
243 smb_sdrc_t smb_nt_transact_ioctl(smb_request_t *, smb_xa_t *);
244 smb_sdrc_t smb_nt_transact_rename(smb_request_t *, smb_xa_t *);
245 smb_sdrc_t smb_nt_transact_query_quota(smb_request_t *, smb_xa_t *);
246 smb_sdrc_t smb_nt_transact_set_quota(smb_request_t *, smb_xa_t *);

248 smb_sdrc_t smb_com_trans2_open2(smb_request_t *, smb_xa_t *);
249 smb_sdrc_t smb_com_trans2_create_directory(smb_request_t *, smb_xa_t *);
250 smb_sdrc_t smb_com_trans2_find_first2(smb_request_t *, smb_xa_t *);
```

```
251 smb_sdrc_t smb_com_trans2_find_next2(smb_request_t *, smb_xa_t *);
252 smb_sdrc_t smb_com_trans2_query_fs_information(smb_request_t *, smb_xa_t *);
253 smb_sdrc_t smb_com_trans2_set_fs_information(smb_request_t *, smb_xa_t *);
254 smb_sdrc_t smb_com_trans2_query_path_information(smb_request_t *, smb_xa_t *);
255 smb_sdrc_t smb_com_trans2_query_file_information(smb_request_t *, smb_xa_t *);
256 smb_sdrc_t smb_com_trans2_set_path_information(smb_request_t *, smb_xa_t *);
257 smb_sdrc_t smb_com_trans2_set_file_information(smb_request_t *, smb_xa_t *);
258 smb_sdrc_t smb_com_trans2_get_dfs_referral(smb_request_t *, smb_xa_t *);

260 uint32_t smb_query_stream_info(smb_request_t *, mbuf_chain_t *,
261     smb_queryinfo_t *);
262 int smb_fssize(smb_request_t *, smb_fssize_t *);

264 /* smb_quota.c */
265 uint32_t smb_quota_query_user_quota(smb_request_t *, uid_t, smb_quota_t *);
266 int smb_quota_query(smb_server_t *, smb_quota_query_t *,
267     smb_quota_response_t *);
268 int smb_quota_set(smb_server_t *, smb_quota_set_t *, uint32_t *);
269 uint32_t smb_quota_init_sids(mbuf_chain_t *, smb_quota_query_t *,
270     smb_ofile_t *);
271 uint32_t smb_quota_decode_sids(mbuf_chain_t *, list_t *);
272 void smb_quota_free_sids(smb_quota_query_t *);
273 void smb_quota_max_quota(mbuf_chain_t *, smb_quota_query_t *);
274 uint32_t smb_quota_decode_quotas(mbuf_chain_t *, list_t *);
275 uint32_t smb_quota_encode_quotas(mbuf_chain_t *, smb_quota_query_t *,
276     smb_quota_response_t *, smb_ofile_t *);
277 void smb_quota_free_quotas(list_t *);

279 void smb_query_shortname(smb_node_t *, smb_queryinfo_t *);

281 uint32_t smb_dfs_fsctl(smb_request_t *, smb_fsctl_t *);
282 uint32_t smb_dfs_get_referrals(smb_request_t *, smb_fsctl_t *);

284 int smb1_newrq_negotiate(smb_request_t *);
285 smb_sdrc_t smb1_negotiate_smb2(smb_request_t *sr);

287 uint32_t smb_set_basic_info(smb_request_t *, smb_setinfo_t *);
288 uint32_t smb_set_eof_info(smb_request_t *, smb_setinfo_t *);
289 uint32_t smb_set_alloc_info(smb_request_t *, smb_setinfo_t *);
290 uint32_t smb_set_disposition_info(smb_request_t *, smb_setinfo_t *);

292 uint32_t smb_setinfo_rename(smb_request_t *, smb_node_t *, char *, int);
293 uint32_t smb_common_rename(smb_request_t *, smb_fqi_t *, smb_fqi_t *);

295 uint32_t smb_setinfo_link(smb_request_t *, smb_node_t *, char *, int);
296 uint32_t smb_make_link(smb_request_t *, smb_fqi_t *, smb_fqi_t *);


299 /*
300  * Logging functions
301  */
302 void smb_log_flush(void);
303 void smb_correct_keep_alive_values(uint32_t new_keep_alive);
304 void smb_close_all_connections(void);

306 int smb_net_id(uint32_t);

308 /*
309  * oplock functions - node operations
310  */
311 void smb_oplock_acquire(smb_request_t *sr, smb_node_t *, smb_ofile_t *);
312 void smb_oplock_release(smb_node_t *, smb_ofile_t *);
313 int smb_oplock_break(smb_request_t *, smb_node_t *, uint32_t);
314 void smb_oplock_break_levelII(smb_node_t *);
315 void smb_oplock_ack(smb_node_t *, smb_ofile_t *, uint8_t);
316 void smb_oplock_broadcast(smb_node_t *);
```



```
318 void smb1_oplock_break_notification(smb_request_t *, uint8_t);
319 void smb2_oplock_break_notification(smb_request_t *, uint8_t);

321 /*
322  * range lock functions - node operations
323  */
324 uint32_t smb_lock_get_lock_count(smb_node_t *, smb_ofile_t *);
325 uint32_t smb_unlock_range(smb_request_t *, uint64_t, uint64_t, uint32_t);
326 uint32_t smb_lock_range(smb_request_t *, uint64_t, uint64_t, uint32_t,
327     uint32_t, uint32_t);
328 uint32_t smb_lock_range_cancel(smb_request_t *, uint64_t, uint64_t, uint32_t);
329 void smb_lock_range_error(smb_request_t *, uint32_t);

331 int smb_lock_range_access(smb_request_t *, smb_node_t *,
332     uint64_t, uint64_t, boolean_t);

334 DWORD smb_nbl_conflict(smb_node_t *, uint64_t, uint64_t, nbl_op_t);

336 void smb_mangle(const char *, ino64_t, char *, size_t);
337 int smb_unmangle(smb_node_t *, char *, char *, int, uint32_t);
338 boolean_t smb_needs_mangled(const char *);
339 boolean_t smb_maybe_mangled(char *);
340 boolean_t smb_is_reserved_dos_name(const char *);
341 boolean_t smb_is_invalid_filename(const char *);

343 void smbsr_cleanup(smb_request_t *sr);

345 int smbsr_connect_tree(smb_request_t *);

347 int smb_common_create_directory(smb_request_t *);

349 void    smb_convert_wildcards(char *);
350 boolean_t smb_contains_wildcards(const char *);
351 int     smb_ascii_or_unicode_strlen(smb_request_t *, char *);
352 int     smb_ascii_or_unicode_strlen_null(smb_request_t *, char *);
353 int     smb_ascii_or_unicode_null_len(smb_request_t *);

355 int     smb_search(smb_request_t *);

357 uint32_t smb_common_create(smb_request_t *);
358 uint32_t smb_common_open(smb_request_t *);

360 int smb_common_write(smb_request_t *, smb_rw_param_t *);

362 void smb_pathname_init(smb_request_t *, smb_pathname_t *, char *);
363 boolean_t smb_pathname_validate(smb_request_t *, smb_pathname_t *);
364 boolean_t smb_validate_dirname(smb_request_t *, smb_pathname_t *);
365 boolean_t smb_validate_object_name(smb_request_t *, smb_pathname_t *);
366 boolean_t smb_validate_stream_name(smb_request_t *, smb_pathname_t *);
367 boolean_t smb_is_stream_name(char *);
368 void smb_stream_parse_name(char *, char *, char *);


371 uint32_t smb_omode_to_amask(uint32_t desired_access);

373 void    sshow_distribution_info(char *);

375 uint32_t smb2sr_go_async(smb_request_t *sr,
376         smb_sdrc_t (*async_func)(smb_request_t *));

378 void    smb_dispatch_stats_init(smb_server_t *);
379 void    smb_dispatch_stats_fini(smb_server_t *);
380 void    smb_dispatch_stats_update(smb_server_t *,
381                 smb_kstat_req_t *, int, int);
```

```
383 int     smb1sr_newrq(smb_request_t *);
384 int     smb1sr_newrq_cancel(smb_request_t *);
385 void    smb1sr_work(smb_request_t *);

387 int     smbsr_encode_empty_result(smb_request_t *);
388 void    smbsr_lookup_file(smb_request_t *);
389 void    smbsr_release_file(smb_request_t *);

391 int     smbsr_decode_vwv(smb_request_t *sr, const char *, ...);
392 int     smbsr_decode_data(smb_request_t *sr, const char *, ...);
393 boolean_t smbsr_decode_data_avail(smb_request_t *);
394 int     smbsr_encode_result(smb_request_t *, int, int, const char *, ...);
395 smb_xa_t *smbsr_lookup_xa(smb_request_t *sr);
396 void    smbsr_send_reply(smb_request_t *);

398 uint32_t smb_errno2status(int);
399 uint16_t smb_status2doserr(uint32_t);
400 void    smbsr_map_errno(int, smb_error_t *);
401 void    smbsr_set_error(smb_request_t *, smb_error_t *);
402 void    smbsr_errno(smb_request_t *, int);
403 void    smbsr_status(smb_request_t *, DWORD, uint16_t, uint16_t);
404 #define smbsr_error(SR, ST, CL, CO) \
405         smbsr_status(SR, ST, CL, CO)
406 #define smbsr_warn(SR, ST, CL, CO) \
407         smbsr_status(SR, ST, CL, CO)
408 void    smbsr_status_smb2(smb_request_t *, DWORD);

410 int     clock_get_milli_uptime(void);

412 int     smb_mbc_vencodef(mbuf_chain_t *, const char *, va_list);
413 int     smb_mbc_vdecodef(mbuf_chain_t *, const char *, va_list);
414 int     smb_mbc_decodef(mbuf_chain_t *, const char *, ...);
415 int     smb_mbc_encodef(mbuf_chain_t *, const char *, ...);
416 int     smb_mbc_peek(mbuf_chain_t *, int, const char *, ...);
417 int     smb_mbc_poke(mbuf_chain_t *, int, const char *, ...);
418 int     smb_mbc_put_mem(mbuf_chain_t *, void *, int);
419 int     smb_mbc_copy(mbuf_chain_t *, const mbuf_chain_t *, int, int);

421 void    smbsr_encode_header(smb_request_t *sr, int wct,
422                     int bcc, const char *fmt, ...);

424 void smb_encode_sd(mbuf_chain_t *, smb_sd_t *, uint32_t);
425 void smb_encode_sid(mbuf_chain_t *, smb_sid_t *);
426 smb_sid_t *smb_decode_sid(mbuf_chain_t *, uint32_t);
427 uint32_t smb_decode_sd(mbuf_chain_t *, smb_sd_t *);

429 uint32_t smb_pad_align(uint32_t, uint32_t);

431 /*
432  * Socket functions
433  */
434 ksocket_t smb_socreate(int domain, int type, int protocol);
435 void smb_soshutdown(ksocket_t so);
436 void smb_sodestroy(ksocket_t so);
437 int smb_sorecv(ksocket_t so, void *msg, size_t len);
438 void smb_net_txl_constructor(smb_txlst_t *);
439 void smb_net_txl_destructor(smb_txlst_t *);
440 int smb_net_send_uio(smb_session_t *, struct uio *);

442 /*
443  * SMB RPC interface
444  */
445 void smb_opipe_dealloc(smb_opipe_t *);
446 int smb_opipe_open(smb_request_t *, uint32_t);
447 void smb_opipe_close(smb_ofile_t *);
448 int smb_opipe_read(smb_request_t *, struct uio *);
```



```
449 int smb_opipe_write(smb_request_t *, struct uio *);
450 int smb_opipe_getattr(smb_ofile_t *, smb_attr_t *);
451 int smb_opipe_getname(smb_ofile_t *, char *, size_t);
452 uint32_t smb_opipe_fsctl(smb_request_t *, smb_fsctl_t *);
453 uint32_t smb_opipe_transceive(smb_request_t *, smb_fsctl_t *);

455 void smb_kdoor_init(smb_server_t *);
456 void smb_kdoor_fini(smb_server_t *);
457 int smb_kdoor_open(smb_server_t *, int);
458 void smb_kdoor_close(smb_server_t *);
459 int smb_kdoor_upcall(smb_server_t *, uint32_t,
460         void *, xdrproc_t, void *, xdrproc_t);
461 void fksmb_kdoor_open(smb_server_t *, void *);

463 /*
464  * SMB server functions (file smb_server.c)
465  */
466 int smb_server_get_count(void);
467 int smb_server_g_init(void);
468 void smb_server_g_fini(void);
469 int smb_server_create(void);
470 int smb_server_delete(void);
471 int smb_server_configure(smb_ioc_cfg_t *);
472 int smb_server_start(smb_ioc_start_t *);
473 int smb_server_stop(void);
474 boolean_t smb_server_is_stopping(smb_server_t *);
475 void smb_server_cancel_event(smb_server_t *, uint32_t);
476 int smb_server_notify_event(smb_ioc_event_t *);
477 uint32_t smb_server_get_session_count(smb_server_t *);
478 int smb_server_set_gmtoff(smb_ioc_gmt_t *);
479 int smb_server_numopen(smb_ioc_opennum_t *);
480 int smb_server_enum(smb_ioc_svcenum_t *);
481 int smb_server_session_close(smb_ioc_session_t *);
482 int smb_server_file_close(smb_ioc_fileid_t *);
483 int smb_server_sharevp(smb_server_t *, const char *, vnode_t **);
484 int smb_server_unshare(const char *);

486 smb_user_t *smb_server_lookup_ssnid(smb_server_t *, uint64_t);

488 void smb_server_get_cfg(smb_server_t *, smb_kmod_cfg_t *);

490 int smb_server_spooldoc(smb_ioc_spooldoc_t *);
491 int smb_spool_add_doc(smb_tree_t *, smb_kspooldoc_t *);
492 void smb_spool_add_fid(smb_server_t *, uint16_t);

494 void smb_server_inc_nbt_sess(smb_server_t *);
495 void smb_server_dec_nbt_sess(smb_server_t *);
496 void smb_server_inc_tcp_sess(smb_server_t *);
497 void smb_server_dec_tcp_sess(smb_server_t *);
498 void smb_server_inc_users(smb_server_t *);
499 void smb_server_dec_users(smb_server_t *);
500 void smb_server_inc_trees(smb_server_t *);
501 void smb_server_dec_trees(smb_server_t *);
502 void smb_server_inc_files(smb_server_t *);
503 void smb_server_dec_files(smb_server_t *);
504 void smb_server_inc_pipes(smb_server_t *);
505 void smb_server_dec_pipes(smb_server_t *);
506 void smb_server_add_rxb(smb_server_t *, int64_t);
507 void smb_server_add_txb(smb_server_t *, int64_t);
508 void smb_server_inc_req(smb_server_t *);

510 void smb_server_post_session(smb_session_t *);

512 smb_event_t *smb_event_create(smb_server_t *, int);
513 void smb_event_destroy(smb_event_t *);
514 uint32_t smb_event_txid(smb_event_t *);
```

```
515 int smb_event_wait(smb_event_t *);
516 void smb_event_notify(smb_server_t *, uint32_t);

518 /*
519  * SMB node functions (file smb_node.c)
520  */
521 void smb_node_init(void);
522 void smb_node_fini(void);
523 smb_node_t *smb_node_lookup(smb_request_t *, smb_arg_open_t *,
524     cred_t *, vnode_t *, char *, smb_node_t *, smb_node_t *);
525 smb_node_t *smb_stream_node_lookup(smb_request_t *, cred_t *,
526     smb_node_t *, vnode_t *, vnode_t *, char *);

528 void smb_node_ref(smb_node_t *);
529 void smb_node_release(smb_node_t *);
530 void smb_node_rename(smb_node_t *, smb_node_t *, smb_node_t *, char *);
531 int smb_node_root_init(smb_server_t *, smb_node_t **);
532 void smb_node_add_lock(smb_node_t *, smb_lock_t *);
533 void smb_node_destroy_lock(smb_node_t *, smb_lock_t *);
534 void smb_node_destroy_lock_by_ofile(smb_node_t *, smb_ofile_t *);
535 void smb_node_start_crit(smb_node_t *, krw_t);
536 void smb_node_end_crit(smb_node_t *);
537 int smb_node_in_crit(smb_node_t *);
538 void smb_node_rdlock(smb_node_t *);
539 void smb_node_wrlock(smb_node_t *);
540 void smb_node_unlock(smb_node_t *);
541 void smb_node_add_ofile(smb_node_t *, smb_ofile_t *);
542 void smb_node_rem_ofile(smb_node_t *, smb_ofile_t *);
543 void smb_node_inc_open_ofiles(smb_node_t *);
544 uint32_t smb_node_dec_open_ofiles(smb_node_t *);
545 void smb_node_inc_opening_count(smb_node_t *);
546 void smb_node_dec_opening_count(smb_node_t *);
547 boolean_t smb_node_is_file(smb_node_t *);
548 boolean_t smb_node_is_dir(smb_node_t *);
549 boolean_t smb_node_is_symlink(smb_node_t *);
550 boolean_t smb_node_is_dfslink(smb_node_t *);
551 boolean_t smb_node_is_reparse(smb_node_t *);
552 boolean_t smb_node_is_vfsroot(smb_node_t *);
553 boolean_t smb_node_is_system(smb_node_t *);

555 uint32_t smb_node_open_check(smb_node_t *, uint32_t, uint32_t);
556 DWORD smb_node_rename_check(smb_node_t *);
557 DWORD smb_node_delete_check(smb_node_t *);
558 boolean_t smb_node_share_check(smb_node_t *);

560 void smb_node_fcn_subscribe(smb_node_t *);
561 void smb_node_fcn_unsubscribe(smb_node_t *);
562 void smb_node_notify_change(smb_node_t *, uint_t, const char *);

564 int smb_node_getattr(smb_request_t *, smb_node_t *, cred_t *,
565     smb_ofile_t *, smb_attr_t *);
566 int smb_node_setattr(smb_request_t *, smb_node_t *, cred_t *,
567     smb_ofile_t *, smb_attr_t *);
568 uint32_t smb_node_set_delete_on_close(smb_node_t *, cred_t *, uint32_t);
569 void smb_node_reset_delete_on_close(smb_node_t *);
570 void smb_node_delete_on_close(smb_node_t *);
571 boolean_t smb_node_file_is_readonly(smb_node_t *);
572 int smb_node_getpath(smb_node_t *, vnode_t *, char *, uint32_t);
573 int smb_node_getmntpath(smb_node_t *, char *, uint32_t);
574 int smb_node_getshrpath(smb_node_t *, smb_tree_t *, char *, uint32_t);

576 /*
577  * Pathname functions
578  */

580 int smb_pathname_reduce(smb_request_t *, cred_t *,
```



```
581     const char *, smb_node_t *, smb_node_t *, smb_node_t **, char *);

583 int smb_pathname(smb_request_t *, char *, int, smb_node_t *,
584     smb_node_t *, smb_node_t **, smb_node_t **, cred_t *);

586 /*
587  * smb_vfs functions
588  */

590 int smb_vfs_hold(smb_export_t *, vfs_t *);
591 void smb_vfs_rele(smb_export_t *, vfs_t *);
592 void smb_vfs_rele_all(smb_export_t *);

594 /*
595  * smb_notify.c
596  */
597 uint32_t smb_notify_act1(smb_request_t *, uint32_t, uint32_t);
598 uint32_t smb_notify_act2(smb_request_t *);
599 uint32_t smb_notify_act3(smb_request_t *);
600 void smb_notify_ofile(smb_ofile_t *, uint_t, const char *);
601 void smb_nt_transact_notify_finish(void *);
602 void smb2_change_notify_finish(void *);

604 int smb_fem_fcn_install(smb_node_t *);
605 void smb_fem_fcn_uninstall(smb_node_t *);
606 int smb_fem_oplock_install(smb_node_t *);
607 void smb_fem_oplock_uninstall(smb_node_t *);

609 /* FEM */

611 int smb_fem_init(void);
612 void smb_fem_fini(void);

614 int smb_try_grow(smb_request_t *sr, int64_t new_size);

616 unsigned short smb_worker_getnum();

618 /* SMB signing routines smb_signing.c */
619 int smb_sign_begin(smb_request_t *, smb_token_t *);
620 int smb_sign_check_request(smb_request_t *);
621 int smb_sign_check_secondary(smb_request_t *, unsigned int);
622 void smb_sign_reply(smb_request_t *, mbuf_chain_t *);
623 /* SMB2, but here because it's called from common code. */
624 int smb2_sign_begin(smb_request_t *, smb_token_t *);
625 int smb3_encrypt_begin(smb_request_t *, smb_token_t *);
626 void smb3_encrypt_fini(smb_session_t *);

628 boolean_t smb_sattr_check(uint16_t, uint16_t);

630 void smb_request_cancel(smb_request_t *);
631 void smb_request_wait(smb_request_t *);

633 /*
634  * authentication support (smb_authenticate.c)
635  */
636 int smb_authenticate_ext(smb_request_t *);
637 int smb_authenticate_old(smb_request_t *);
638 void smb_authsock_close(smb_user_t *, ksocket_t);

640 /*
641  * session functions (file smb_session.c)
642  */
643 smb_session_t *smb_session_create(ksocket_t, uint16_t, smb_server_t *, int);
644 void smb_session_hold(smb_session_t *);
645 void smb_session_release(smb_session_t *);
646 void smb_session_receiver(smb_session_t *);
```

```
647 void smb_session_disconnect(smb_session_t *);
648 void smb_session_timers(smb_server_t *);
649 void smb_session_delete(smb_session_t *session);
650 void smb_session_cancel_requests(smb_session_t *, smb_tree_t *,
651     smb_request_t *);
652 void smb_session_config(smb_session_t *session);
653 void smb_session_disconnect_from_share(smb_llist_t *, char *);
654 smb_user_t *smb_session_dup_user(smb_session_t *, char *, char *);
655 smb_user_t *smb_session_lookup_ssnid(smb_session_t *, uint64_t);
656 smb_user_t *smb_session_lookup_uid(smb_session_t *, uint16_t);
657 smb_user_t *smb_session_lookup_uid_st(smb_session_t *,
658     uint64_t, uint16_t, smb_user_state_t);
659 void smb_session_post_user(smb_session_t *, smb_user_t *);
660 void smb_session_post_tree(smb_session_t *, smb_tree_t *);
661 smb_tree_t *smb_session_lookup_tree(smb_session_t *, uint16_t);
662 smb_tree_t *smb_session_lookup_share(smb_session_t *, const char *,
663     smb_tree_t *);
664 smb_tree_t *smb_session_lookup_volume(smb_session_t *, const char *,
665     smb_tree_t *);
666 void smb_session_close_pid(smb_session_t *, uint32_t);
667 void smb_session_disconnect_owned_trees(smb_session_t *, smb_user_t *);
668 void smb_session_disconnect_trees(smb_session_t *);
669 void smb_session_disconnect_share(smb_session_t *, const char *);
670 void smb_session_getclient(smb_session_t *, char *, size_t);
671 boolean_t smb_session_isclient(smb_session_t *, const char *);
672 void smb_session_correct_keep_alive_values(smb_llist_t *, uint32_t);
673 int smb_session_send(smb_session_t *, uint8_t type, mbuf_chain_t *);
674 int smb_session_xprt_gethdr(smb_session_t *, smb_xprt_t *);
675 boolean_t smb_session_oplocks_enable(smb_session_t *);
676 boolean_t smb_session_levelII_oplocks(smb_session_t *);

678 #define SMB_SESSION_GET_ID(s)   ((s)->s_kid)

680 smb_request_t *smb_request_alloc(smb_session_t *, int);
681 void smb_request_free(smb_request_t *);

683 /*
684  * ofile functions (file smb_ofile.c)
685  */
686 smb_ofile_t *smb_ofile_lookup_by_fid(smb_request_t *, uint16_t);
687 smb_ofile_t *smb_ofile_lookup_by_uniqid(smb_tree_t *, uint32_t);
688 smb_ofile_t *smb_ofile_lookup_by_persistid(smb_request_t *, uint64_t);
689 boolean_t smb_ofile_disallow_fclose(smb_ofile_t *);
690 smb_ofile_t *smb_ofile_open(smb_request_t *, smb_node_t *,
691     smb_arg_open_t *, uint16_t, uint32_t, smb_error_t *);
692 void smb_ofile_close(smb_ofile_t *, int32_t);
693 void smb_ofile_delete(void *);
694 uint32_t smb_ofile_access(smb_ofile_t *, cred_t *, uint32_t);
695 int smb_ofile_seek(smb_ofile_t *, ushort_t, int32_t, uint32_t *);
696 void smb_ofile_flush(smb_request_t *, smb_ofile_t *);
697 boolean_t smb_ofile_hold(smb_ofile_t *);
698 void smb_ofile_release(smb_ofile_t *);
699 void smb_ofile_request_complete(smb_ofile_t *);
700 void smb_ofile_close_all(smb_tree_t *);
701 void smb_ofile_close_all_by_pid(smb_tree_t *, uint16_t);
702 void smb_ofile_set_flags(smb_ofile_t *, uint32_t);
703 boolean_t smb_ofile_is_open(smb_ofile_t *);
704 int smb_ofile_enum(smb_ofile_t *, smb_svcenum_t *);
705 uint32_t smb_ofile_open_check(smb_ofile_t *, uint32_t, uint32_t);
706 uint32_t smb_ofile_rename_check(smb_ofile_t *);
707 uint32_t smb_ofile_delete_check(smb_ofile_t *);
708 boolean_t smb_ofile_share_check(smb_ofile_t *);
709 cred_t *smb_ofile_getcred(smb_ofile_t *);
710 void smb_ofile_set_delete_on_close(smb_ofile_t *);
711 void smb_delayed_write_timer(smb_llist_t *);
712 void smb_ofile_set_quota_resume(smb_ofile_t *, char *);
```



```
713 void smb_ofile_get_quota_resume(smb_ofile_t *, char *, int);
714 uint32_t smb_ofile_set_resilient(smb_request_t *, smb_fsctl_t *);

716 #define SMB_OFILE_GET_SESSION(of)       ((of)->f_session)
717 #define SMB_OFILE_GET_TREE(of)          ((of)->f_tree)
718 #define SMB_OFILE_GET_FID(of)           ((of)->f_fid)
719 #define SMB_OFILE_GET_NODE(of)          ((of)->f_node)

721 #define smb_ofile_granted_access(_of_)  ((_of_)->f_granted_access)

723 /*
724  * odir functions (file smb_odir.c)
725  */
726 uint32_t smb_odir_openpath(smb_request_t *, char *, uint16_t, uint32_t,
727     smb_odir_t **);
728 uint32_t smb_odir_openfh(smb_request_t *, const char *, uint16_t,
729     smb_odir_t **);
730 uint32_t smb_odir_openat(smb_request_t *, smb_node_t *, smb_odir_t **);
731 void smb_odir_reopen(smb_odir_t *, const char *, uint16_t);
732 void smb_odir_close(smb_odir_t *);
733 boolean_t smb_odir_hold(smb_odir_t *);
734 void smb_odir_release(smb_odir_t *);
735 void smb_odir_delete(void *);

737 int smb_odir_read(smb_request_t *, smb_odir_t *,
738     smb_odirent_t *, boolean_t *);
739 int smb_odir_read_fileinfo(smb_request_t *, smb_odir_t *,
740     smb_fileinfo_t *, uint16_t *);
741 int smb_odir_read_streaminfo(smb_request_t *, smb_odir_t *,
742     smb_streaminfo_t *, boolean_t *);

744 void smb_odir_save_cookie(smb_odir_t *, int, uint32_t cookie);
745 void smb_odir_save_fname(smb_odir_t *, uint32_t, const char *);

747 void smb_odir_resume_at(smb_odir_t *, smb_odir_resume_t *);

749 /*
750  * SMB user functions (file smb_user.c)
751  */
752 smb_user_t *smb_user_new(smb_session_t *);
753 int smb_user_logon(smb_user_t *, cred_t *,
754     char *, char *, uint32_t, uint32_t, uint32_t);
755 void smb_user_logoff(smb_user_t *);
756 void smb_user_delete(void *);
757 boolean_t smb_user_is_admin(smb_user_t *);
758 boolean_t smb_user_namecmp(smb_user_t *, const char *);
759 int smb_user_enum(smb_user_t *, smb_svcenum_t *);
760 boolean_t smb_user_hold(smb_user_t *);
761 void smb_user_hold_internal(smb_user_t *);
762 void smb_user_release(smb_user_t *);
763 cred_t *smb_user_getcred(smb_user_t *);
764 cred_t *smb_user_getprivcred(smb_user_t *);
765 void smb_user_netinfo_init(smb_user_t *, smb_netuserinfo_t *);
766 void smb_user_netinfo_fini(smb_netuserinfo_t *);
767 int smb_user_netinfo_encode(smb_user_t *, uint8_t *, size_t, uint32_t *);
768 smb_token_t *smb_get_token(smb_session_t *, smb_logon_t *);
769 cred_t *smb_cred_create(smb_token_t *);
770 void smb_user_setcred(smb_user_t *, cred_t *, uint32_t);
771 boolean_t smb_is_same_user(smb_user_t *, smb_user_t *);

773 /*
774  * SMB tree functions (file smb_tree.c)
775  */
776 uint32_t smb_tree_connect(smb_request_t *);
777 void smb_tree_disconnect(smb_tree_t *, boolean_t);
778 void smb_tree_dealloc(void *);
```

```
779 void smb_tree_post_ofile(smb_tree_t *, smb_ofile_t *);
780 void smb_tree_post_odir(smb_tree_t *, smb_odir_t *);
781 void smb_tree_close_pid(smb_tree_t *, uint32_t);
782 boolean_t smb_tree_has_feature(smb_tree_t *, uint_t);
783 int smb_tree_enum(smb_tree_t *, smb_svcenum_t *);
784 int smb_tree_fclose(smb_tree_t *, uint32_t);
785 boolean_t smb_tree_hold(smb_tree_t *);
786 void smb_tree_hold_internal(smb_tree_t *);
787 void smb_tree_release(smb_tree_t *);
788 smb_odir_t *smb_tree_lookup_odir(smb_request_t *, uint16_t);
789 boolean_t smb_tree_is_connected(smb_tree_t *);
790 boolean_t smb_tree_is_connected_locked(smb_tree_t *);
791 #define SMB_TREE_GET_TID(tree)          ((tree)->t_tid)

793 smb_xa_t *smb_xa_create(smb_session_t *session, smb_request_t *sr,
794     uint32_t total_parameter_count, uint32_t total_data_count,
795     uint32_t max_parameter_count, uint32_t max_data_count,
796     uint32_t max_setup_count, uint32_t setup_word_count);
797 void smb_xa_delete(smb_xa_t *xa);
798 smb_xa_t *smb_xa_hold(smb_xa_t *xa);
799 void smb_xa_rele(smb_session_t *session, smb_xa_t *xa);
800 int smb_xa_open(smb_xa_t *xa);
801 void smb_xa_close(smb_xa_t *xa);
802 int smb_xa_complete(smb_xa_t *xa);
803 smb_xa_t *smb_xa_find(smb_session_t *session, uint32_t pid, uint16_t mid);

805 struct mbuf *smb_mbuf_get(uchar_t *buf, int nbytes);
806 struct mbuf *smb_mbuf_allocate(struct uio *uio);
807 void smb_mbuf_trim(struct mbuf *mhead, int nbytes);

810 void smb_check_status(void);
811 int smb_handle_write_raw(smb_session_t *session, smb_request_t *sr);

809 int32_t smb_time_gmt_to_local(smb_request_t *, int32_t);
810 int32_t smb_time_local_to_gmt(smb_request_t *, int32_t);
811 int32_t smb_time_dos_to_unix(int16_t, int16_t);
812 void smb_time_unix_to_dos(int32_t, int16_t *, int16_t *);
813 void smb_time_nt_to_unix(uint64_t nt_time, timestruc_t *unix_time);
814 uint64_t smb_time_unix_to_nt(timestruc_t *);

816 int netbios_name_isvalid(char *in, char *out);

818 int uioxfer(struct uio *src_uio, struct uio *dst_uio, int n);

820 /*
821  * Pool ID function prototypes
822  */
823 int     smb_idpool_constructor(smb_idpool_t *pool);
824 void    smb_idpool_destructor(smb_idpool_t  *pool);
825 int     smb_idpool_alloc(smb_idpool_t *pool, uint16_t *id);
826 void    smb_idpool_free(smb_idpool_t *pool, uint16_t id);

828 /*
829  * SMB locked list function prototypes
830  */
831 void    smb_llist_init(void);
832 void    smb_llist_fini(void);
833 void    smb_llist_constructor(smb_llist_t *, size_t, size_t);
834 void    smb_llist_destructor(smb_llist_t *);
835 void    smb_llist_exit(smb_llist_t *);
836 void    smb_llist_post(smb_llist_t *, void *, smb_dtorproc_t);
837 void    smb_llist_flush(smb_llist_t *);
838 void    smb_llist_insert_head(smb_llist_t *ll, void *obj);
839 void    smb_llist_insert_tail(smb_llist_t *ll, void *obj);
840 void    smb_llist_remove(smb_llist_t *ll, void *obj);
841 int     smb_llist_upgrade(smb_llist_t *ll);
```



```
842 uint32_t smb_llist_get_count(smb_llist_t *ll);
843 #define smb_llist_enter(ll, mode)       rw_enter(&(ll)->ll_lock, mode)
844 #define smb_llist_head(ll)              list_head(&(ll)->ll_list)
845 #define smb_llist_next(ll, obj)         list_next(&(ll)->ll_list, obj)
846 int     smb_account_connected(smb_user_t *user);

848 /*
849  * SMB Synchronized list function prototypes
850  */
851 void    smb_slist_constructor(smb_slist_t *, size_t, size_t);
852 void    smb_slist_destructor(smb_slist_t *);
853 void    smb_slist_insert_head(smb_slist_t *sl, void *obj);
854 void    smb_slist_insert_tail(smb_slist_t *sl, void *obj);
855 void    smb_slist_remove(smb_slist_t *sl, void *obj);
856 void    smb_slist_wait_for_empty(smb_slist_t *sl);
857 void    smb_slist_exit(smb_slist_t *sl);
858 uint32_t smb_slist_move_tail(list_t *lst, smb_slist_t *sl);
859 void    smb_slist_obj_move(smb_slist_t *dst, smb_slist_t *src, void *obj);
860 #define smb_slist_enter(sl)             mutex_enter(&(sl)->sl_mutex)
861 #define smb_slist_head(sl)              list_head(&(sl)->sl_list)
862 #define smb_slist_next(sl, obj)         list_next(&(sl)->sl_list, obj)

864 void    smb_rwx_init(smb_rwx_t *rwx);
865 void    smb_rwx_destroy(smb_rwx_t *rwx);
866 #define smb_rwx_rwenter(rwx, mode)      rw_enter(&(rwx)->rwx_lock, mode)
867 void    smb_rwx_rwexit(smb_rwx_t *rwx);
868 int     smb_rwx_rwwait(smb_rwx_t *rwx, clock_t timeout);
869 #define smb_rwx_xenter(rwx)             mutex_enter(&(rwx)->rwx_mutex)
870 #define smb_rwx_xexit(rwx)              mutex_exit(&(rwx)->rwx_mutex)
871 krw_t   smb_rwx_rwupgrade(smb_rwx_t *rwx);
872 void    smb_rwx_rwdowngrade(smb_rwx_t *rwx, krw_t mode);

874 void    smb_thread_init(smb_thread_t *, char *, smb_thread_ep_t,
875                 void *, pri_t);
876 void    smb_thread_destroy(smb_thread_t *);
877 int     smb_thread_start(smb_thread_t *);
878 void    smb_thread_stop(smb_thread_t *);
879 void    smb_thread_signal(smb_thread_t *);
880 boolean_t smb_thread_continue(smb_thread_t *);
881 boolean_t smb_thread_continue_nowait(smb_thread_t *);
882 boolean_t smb_thread_continue_timedwait(smb_thread_t *, int /* seconds */);

884 uint32_t smb_denymode_to_sharemode(uint32_t desired_access, char *fname);
885 uint32_t smb_ofun_to_crdisposition(uint16_t ofun);

887 /* 100's of ns between 1/1/1970 and 1/1/1601 */
888 #define NT_TIME_BIAS    (134774LL * 24LL * 60LL * 60LL * 10000000LL)

890 uint32_t smb_sd_read(smb_request_t *, smb_sd_t *, uint32_t);
891 uint32_t smb_sd_write(smb_request_t *, smb_sd_t *, uint32_t);

893 acl_t *smb_fsacl_inherit(acl_t *, int, int, cred_t *);
894 acl_t *smb_fsacl_merge(acl_t *, acl_t *);
895 void smb_fsacl_split(acl_t *, acl_t **, acl_t **, int);
896 acl_t *smb_fsacl_from_vsa(vsecattr_t *, acl_type_t);
897 int smb_fsacl_to_vsa(acl_t *, vsecattr_t *, int *);

899 boolean_t smb_ace_is_generic(int);
900 boolean_t smb_ace_is_access(int);
901 boolean_t smb_ace_is_audit(int);

903 uint32_t smb_vss_enum_snapshots(smb_request_t *, smb_fsctl_t *);
904 int smb_vss_lookup_nodes(smb_request_t *, smb_node_t *, smb_node_t *,
905     char *, smb_node_t **, smb_node_t **);
906 vnode_t *smb_lookuppathvptovp(smb_request_t *, char *, vnode_t *, vnode_t *);
```

```
 908 void smb_panic(char *, const char *, int);
 909 #pragma does_not_return(smb_panic)
 910 #define SMB_PANIC()     smb_panic(__FILE__, __func__, __LINE__)

 912 void smb_latency_init(smb_latency_t *);
 913 void smb_latency_destroy(smb_latency_t *);
 914 void smb_latency_add_sample(smb_latency_t *, hrtime_t);
 915 void smb_srqueue_init(smb_srqueue_t *);
 916 void smb_srqueue_destroy(smb_srqueue_t *);
 917 void smb_srqueue_waitq_enter(smb_srqueue_t *);
 918 void smb_srqueue_runq_exit(smb_srqueue_t *);
 919 void smb_srqueue_waitq_to_runq(smb_srqueue_t *);
 920 void smb_srqueue_update(smb_srqueue_t *, smb_kstat_utilization_t *);

 922 void *smb_mem_alloc(size_t);
 923 void *smb_mem_zalloc(size_t);
 924 void *smb_mem_realloc(void *, size_t);
 925 void *smb_mem_rezalloc(void *, size_t);
 926 void smb_mem_free(void *);
 927 void smb_mem_zfree(void *);
 928 char *smb_mem_strdup(const char *);
 929 void smb_srm_init(smb_request_t *);
 930 void smb_srm_fini(smb_request_t *);
 931 void *smb_srm_alloc(smb_request_t *, size_t);
 932 void *smb_srm_zalloc(smb_request_t *, size_t);
 933 void *smb_srm_realloc(smb_request_t *, void *, size_t);
 934 void *smb_srm_rezalloc(smb_request_t *, void *, size_t);
 935 char *smb_srm_strdup(smb_request_t *, const char *);

 937 void smb_export_start(smb_server_t *);
 938 void smb_export_stop(smb_server_t *);

 940 #ifdef  _KERNEL
 941 struct __door_handle;
 942 struct __door_handle *smb_kshare_door_init(int);
 943 void smb_kshare_door_fini(struct __door_handle *);
 944 int smb_kshare_upcall(struct __door_handle *, void *, boolean_t);
 945 #endif  /* _KERNEL */

 947 void smb_kshare_g_init(void);
 948 void smb_kshare_g_fini(void);
 949 void smb_kshare_init(smb_server_t *);
 950 void smb_kshare_fini(smb_server_t *);
 951 int smb_kshare_start(smb_server_t *);
 952 void smb_kshare_stop(smb_server_t *);

 954 int smb_kshare_export_list(smb_ioc_share_t *);
 955 int smb_kshare_unexport_list(smb_ioc_share_t *);
 956 int smb_kshare_info(smb_ioc_shareinfo_t *);
 957 void smb_kshare_enum(smb_server_t *, smb_enumshare_info_t *);
 958 smb_kshare_t *smb_kshare_lookup(smb_server_t *, const char *);
 959 void smb_kshare_release(smb_server_t *, smb_kshare_t *);
 960 int smb_kshare_exec(smb_server_t *, smb_shr_execinfo_t *);
 961 uint32_t smb_kshare_hostaccess(smb_kshare_t *, smb_session_t *);


 964 void smb_avl_create(smb_avl_t *, size_t, size_t, const smb_avl_nops_t *);
 965 void smb_avl_destroy(smb_avl_t *);
 966 int smb_avl_add(smb_avl_t *, void *);
 967 void smb_avl_remove(smb_avl_t *, void *);
 968 void *smb_avl_lookup(smb_avl_t *, void *);
 969 void smb_avl_release(smb_avl_t *, void *);
 970 void smb_avl_iterinit(smb_avl_t *, smb_avl_cursor_t *);
 971 void *smb_avl_iterate(smb_avl_t *, smb_avl_cursor_t *);

 973 void smb_threshold_init(smb_cmd_threshold_t *,
```



```
 974     char *, uint_t, uint_t);
 975 void smb_threshold_fini(smb_cmd_threshold_t *);
 976 int smb_threshold_enter(smb_cmd_threshold_t *);
 977 void smb_threshold_exit(smb_cmd_threshold_t *);
 978 void smb_threshold_wake_all(smb_cmd_threshold_t *);

 980 /* SMB hash function prototypes */
 981 smb_hash_t *smb_hash_create(size_t, size_t, uint32_t num_buckets);
 982 void smb_hash_destroy(smb_hash_t *);
 983 void smb_ptrhash_insert(smb_hash_t *, void *);
 984 void smb_ptrhash_remove(smb_hash_t *, void *);
 985 void *smb_ptrhash_find(smb_hash_t *, void *, void *(*)(void *));

 987 #ifdef  __cplusplus
 988 }
```

*unchanged_portion_omitted_*

```
**********************************************************
   16761 Fri Mar 31 14:29:44 2017
new/usr/src/uts/common/sys/sdt.h
NEX-1643 dtrace provider for smbsrv
Also illumos 1841:
 DTrace smb provider was mis-implemented, doesn't exist.
Add back handlers for read/write raw, so that
 legacy dtrace consumers can find the probes.
Kill extra arg in smb_negotiate
Fix missing "done" probe with smb_notify
Add example consumer: smb-trace.d
fix soi_pid
**********************************************************
______unchanged_portion_omitted_

 144 #define DTRACE_SCHED(name)                                              \
 145         DTRACE_PROBE(__sched_##name);

 147 #define DTRACE_SCHED1(name, type1, arg1)                                \
 148         DTRACE_PROBE1(__sched_##name, type1, arg1);

 150 #define DTRACE_SCHED2(name, type1, arg1, type2, arg2)                   \
 151         DTRACE_PROBE2(__sched_##name, type1, arg1, type2, arg2);

 153 #define DTRACE_SCHED3(name, type1, arg1, type2, arg2, type3, arg3)      \
 154         DTRACE_PROBE3(__sched_##name, type1, arg1, type2, arg2, type3, arg3);

 156 #define DTRACE_SCHED4(name, type1, arg1, type2, arg2,                   \
 157     type3, arg3, type4, arg4)                                           \
 158         DTRACE_PROBE4(__sched_##name, type1, arg1, type2, arg2,         \
 159             type3, arg3, type4, arg4);

 161 #define DTRACE_PROC(name)                                               \
 162         DTRACE_PROBE(__proc_##name);

 164 #define DTRACE_PROC1(name, type1, arg1)                                 \
 165         DTRACE_PROBE1(__proc_##name, type1, arg1);

 167 #define DTRACE_PROC2(name, type1, arg1, type2, arg2)                    \
 168         DTRACE_PROBE2(__proc_##name, type1, arg1, type2, arg2);

 170 #define DTRACE_PROC3(name, type1, arg1, type2, arg2, type3, arg3)       \
 171         DTRACE_PROBE3(__proc_##name, type1, arg1, type2, arg2, type3, arg3);

 173 #define DTRACE_PROC4(name, type1, arg1, type2, arg2,                    \
 174     type3, arg3, type4, arg4)                                           \
 175         DTRACE_PROBE4(__proc_##name, type1, arg1, type2, arg2,          \
 176             type3, arg3, type4, arg4);

 178 #define DTRACE_IO(name)                                                 \
 179         DTRACE_PROBE(__io_##name);

 181 #define DTRACE_IO1(name, type1, arg1)                                   \
 182         DTRACE_PROBE1(__io_##name, type1, arg1);

 184 #define DTRACE_IO2(name, type1, arg1, type2, arg2)                      \
 185         DTRACE_PROBE2(__io_##name, type1, arg1, type2, arg2);

 187 #define DTRACE_IO3(name, type1, arg1, type2, arg2, type3, arg3) \
 188         DTRACE_PROBE3(__io_##name, type1, arg1, type2, arg2, type3, arg3);

 190 #define DTRACE_IO4(name, type1, arg1, type2, arg2,                      \
 191     type3, arg3, type4, arg4)                                           \
 192         DTRACE_PROBE4(__io_##name, type1, arg1, type2, arg2,            \
 193             type3, arg3, type4, arg4);
```



```
 195 #define DTRACE_ISCSI_2(name, type1, arg1, type2, arg2)                  \
 196         DTRACE_PROBE2(__iscsi_##name, type1, arg1, type2, arg2);

 198 #define DTRACE_ISCSI_3(name, type1, arg1, type2, arg2, type3, arg3)     \
 199         DTRACE_PROBE3(__iscsi_##name, type1, arg1, type2, arg2, type3, arg3);

 201 #define DTRACE_ISCSI_4(name, type1, arg1, type2, arg2,                  \
 202     type3, arg3, type4, arg4)                                           \
 203         DTRACE_PROBE4(__iscsi_##name, type1, arg1, type2, arg2,         \
 204             type3, arg3, type4, arg4);

 206 #define DTRACE_ISCSI_5(name, type1, arg1, type2, arg2,                  \
 207     type3, arg3, type4, arg4, type5, arg5)                              \
 208         DTRACE_PROBE5(__iscsi_##name, type1, arg1, type2, arg2,         \
 209             type3, arg3, type4, arg4, type5, arg5);

 211 #define DTRACE_ISCSI_6(name, type1, arg1, type2, arg2,                  \
 212     type3, arg3, type4, arg4, type5, arg5, type6, arg6)                 \
 213         DTRACE_PROBE6(__iscsi_##name, type1, arg1, type2, arg2,         \
 214             type3, arg3, type4, arg4, type5, arg5, type6, arg6);

 216 #define DTRACE_ISCSI_7(name, type1, arg1, type2, arg2,                  \
 217     type3, arg3, type4, arg4, type5, arg5, type6, arg6, type7, arg7)    \
 218         DTRACE_PROBE7(__iscsi_##name, type1, arg1, type2, arg2,         \
 219             type3, arg3, type4, arg4, type5, arg5, type6, arg6,         \
 220             type7, arg7);

 222 #define DTRACE_ISCSI_8(name, type1, arg1, type2, arg2,                  \
 223     type3, arg3, type4, arg4, type5, arg5, type6, arg6,                 \
 224     type7, arg7, type8, arg8)                                           \
 225         DTRACE_PROBE8(__iscsi_##name, type1, arg1, type2, arg2,         \
 226             type3, arg3, type4, arg4, type5, arg5, type6, arg6,         \
 227             type7, arg7, type8, arg8);

 229 #define DTRACE_NFSV3_3(name, type1, arg1, type2, arg2,                  \
 230     type3, arg3)                                                        \
 231         DTRACE_PROBE3(__nfsv3_##name, type1, arg1, type2, arg2,         \
 232             type3, arg3);
 233 #define DTRACE_NFSV3_4(name, type1, arg1, type2, arg2,                  \
 234     type3, arg3, type4, arg4)                                           \
 235         DTRACE_PROBE4(__nfsv3_##name, type1, arg1, type2, arg2,         \
 236             type3, arg3, type4, arg4);

 238 #define DTRACE_NFSV4_1(name, type1, arg1) \
 239         DTRACE_PROBE1(__nfsv4_##name, type1, arg1);

 241 #define DTRACE_NFSV4_2(name, type1, arg1, type2, arg2) \
 242         DTRACE_PROBE2(__nfsv4_##name, type1, arg1, type2, arg2);

 244 #define DTRACE_NFSV4_3(name, type1, arg1, type2, arg2, type3, arg3) \
 245         DTRACE_PROBE3(__nfsv4_##name, type1, arg1, type2, arg2, type3, arg3);

 247 #define DTRACE_SMB_1(name, type1, arg1) \
 248         DTRACE_PROBE1(__smb_##name, type1, arg1);

 250 #define DTRACE_SMB_2(name, type1, arg1, type2, arg2) \
 251         DTRACE_PROBE2(__smb_##name, type1, arg1, type2, arg2);

 253 #define DTRACE_SMB2_1(name, type1, arg1) \
 254         DTRACE_PROBE1(__smb2_##name, type1, arg1);

 256 #define DTRACE_SMB2_2(name, type1, arg1, type2, arg2) \
 257         DTRACE_PROBE2(__smb2_##name, type1, arg1, type2, arg2);

 259 #define DTRACE_IP(name)                                                 \
 260         DTRACE_PROBE(__ip_##name);
```

```
262 #define DTRACE_IP1(name, type1, arg1)                                   \
263         DTRACE_PROBE1(__ip_##name, type1, arg1);

265 #define DTRACE_IP2(name, type1, arg1, type2, arg2)                      \
266         DTRACE_PROBE2(__ip_##name, type1, arg1, type2, arg2);

268 #define DTRACE_IP3(name, type1, arg1, type2, arg2, type3, arg3) \
269         DTRACE_PROBE3(__ip_##name, type1, arg1, type2, arg2, type3, arg3);

271 #define DTRACE_IP4(name, type1, arg1, type2, arg2,                      \
272     type3, arg3, type4, arg4)                                           \
273         DTRACE_PROBE4(__ip_##name, type1, arg1, type2, arg2,            \
274             type3, arg3, type4, arg4);

276 #define DTRACE_IP5(name, type1, arg1, type2, arg2,                      \
277     type3, arg3, type4, arg4, type5, arg5)                              \
278         DTRACE_PROBE5(__ip_##name, type1, arg1, type2, arg2,            \
279             type3, arg3, type4, arg4, type5, arg5);

281 #define DTRACE_IP6(name, type1, arg1, type2, arg2,                      \
282     type3, arg3, type4, arg4, type5, arg6)                              \
283         DTRACE_PROBE6(__ip_##name, type1, arg1, type2, arg2,            \
284             type3, arg3, type4, arg4, type5, arg5, type6, arg6);

286 #define DTRACE_IP7(name, type1, arg1, type2, arg2, type3, arg3,         \
287     type4, arg4, type5, arg5, type6, arg6, type7, arg7)                 \
288         DTRACE_PROBE7(__ip_##name, type1, arg1, type2, arg2,            \
289             type3, arg3, type4, arg4, type5, arg5, type6, arg6,         \
290             type7, arg7);

292 #define DTRACE_TCP(name)                                                \
293         DTRACE_PROBE(__tcp_##name);

295 #define DTRACE_TCP1(name, type1, arg1)                                  \
296         DTRACE_PROBE1(__tcp_##name, type1, arg1);

298 #define DTRACE_TCP2(name, type1, arg1, type2, arg2)                     \
299         DTRACE_PROBE2(__tcp_##name, type1, arg1, type2, arg2);

301 #define DTRACE_TCP3(name, type1, arg1, type2, arg2, type3, arg3)        \
302         DTRACE_PROBE3(__tcp_##name, type1, arg1, type2, arg2, type3, arg3);

304 #define DTRACE_TCP4(name, type1, arg1, type2, arg2,                     \
305     type3, arg3, type4, arg4)                                           \
306         DTRACE_PROBE4(__tcp_##name, type1, arg1, type2, arg2,           \
307             type3, arg3, type4, arg4);

309 #define DTRACE_TCP5(name, type1, arg1, type2, arg2,                     \
310     type3, arg3, type4, arg4, type5, arg5)                              \
311         DTRACE_PROBE5(__tcp_##name, type1, arg1, type2, arg2,           \
312             type3, arg3, type4, arg4, type5, arg5);

314 #define DTRACE_TCP6(name, type1, arg1, type2, arg2,                     \
315     type3, arg3, type4, arg4, type5, arg5, type6, arg6)                 \
316         DTRACE_PROBE6(__tcp_##name, type1, arg1, type2, arg2,           \
317             type3, arg3, type4, arg4, type5, arg5, type6, arg6);

319 #define DTRACE_UDP(name)                                                \
320         DTRACE_PROBE(__udp_##name);

322 #define DTRACE_UDP1(name, type1, arg1)                                  \
323         DTRACE_PROBE1(__udp_##name, type1, arg1);

325 #define DTRACE_UDP2(name, type1, arg1, type2, arg2)                     \
326         DTRACE_PROBE2(__udp_##name, type1, arg1, type2, arg2);
```



```
328 #define DTRACE_UDP3(name, type1, arg1, type2, arg2, type3, arg3)        \
329         DTRACE_PROBE3(__udp_##name, type1, arg1, type2, arg2, type3, arg3);

331 #define DTRACE_UDP4(name, type1, arg1, type2, arg2,                     \
332     type3, arg3, type4, arg4)                                           \
333         DTRACE_PROBE4(__udp_##name, type1, arg1, type2, arg2,           \
334             type3, arg3, type4, arg4);

336 #define DTRACE_UDP5(name, type1, arg1, type2, arg2,                     \
337     type3, arg3, type4, arg4, type5, arg5)                              \
338         DTRACE_PROBE5(__udp_##name, type1, arg1, type2, arg2,           \
339             type3, arg3, type4, arg4, type5, arg5);


342 #define DTRACE_SYSEVENT2(name, type1, arg1, type2, arg2)                \
343         DTRACE_PROBE2(__sysevent_##name, type1, arg1, type2, arg2);

345 #define DTRACE_XPV(name)                                                \
346         DTRACE_PROBE(__xpv_##name);

348 #define DTRACE_XPV1(name, type1, arg1)                                  \
349         DTRACE_PROBE1(__xpv_##name, type1, arg1);

351 #define DTRACE_XPV2(name, type1, arg1, type2, arg2)                     \
352         DTRACE_PROBE2(__xpv_##name, type1, arg1, type2, arg2);

354 #define DTRACE_XPV3(name, type1, arg1, type2, arg2, type3, arg3)        \
355         DTRACE_PROBE3(__xpv_##name, type1, arg1, type2, arg2, type3, arg3);

357 #define DTRACE_XPV4(name, type1, arg1, type2, arg2, type3, arg3,        \
358             type4, arg4)                                                \
359         DTRACE_PROBE4(__xpv_##name, type1, arg1, type2, arg2,           \
360             type3, arg3, type4, arg4);

362 #define DTRACE_FC_1(name, type1, arg1) \
363         DTRACE_PROBE1(__fc_##name, type1, arg1);

365 #define DTRACE_FC_2(name, type1, arg1, type2, arg2) \
366         DTRACE_PROBE2(__fc_##name, type1, arg1, type2, arg2);

368 #define DTRACE_FC_3(name, type1, arg1, type2, arg2, type3, arg3) \
369         DTRACE_PROBE3(__fc_##name, type1, arg1, type2, arg2, type3, arg3);

371 #define DTRACE_FC_4(name, type1, arg1, type2, arg2, type3, arg3, type4, arg4) \
372         DTRACE_PROBE4(__fc_##name, type1, arg1, type2, arg2, type3, arg3, \
373             type4, arg4);

375 #define DTRACE_FC_5(name, type1, arg1, type2, arg2, type3, arg3,        \
376             type4, arg4, type5, arg5)                                   \
377         DTRACE_PROBE5(__fc_##name, type1, arg1, type2, arg2, type3, arg3, \
378             type4, arg4, type5, arg5);

380 #define DTRACE_SRP_1(name, type1, arg1)                                 \
381         DTRACE_PROBE1(__srp_##name, type1, arg1);

383 #define DTRACE_SRP_2(name, type1, arg1, type2, arg2)                    \
384         DTRACE_PROBE2(__srp_##name, type1, arg1, type2, arg2);

386 #define DTRACE_SRP_3(name, type1, arg1, type2, arg2, type3, arg3)       \
387         DTRACE_PROBE3(__srp_##name, type1, arg1, type2, arg2, type3, arg3);

389 #define DTRACE_SRP_4(name, type1, arg1, type2, arg2, type3, arg3,       \
390             type4, arg4)                                                \
391         DTRACE_PROBE4(__srp_##name, type1, arg1, type2, arg2,           \
392             type3, arg3, type4, arg4);
```

```
394 #define DTRACE_SRP_5(name, type1, arg1, type2, arg2, type3, arg3,       \
395             type4, arg4, type5, arg5)                                   \
396         DTRACE_PROBE5(__srp_##name, type1, arg1, type2, arg2,           \
397             type3, arg3, type4, arg4, type5, arg5);

399 #define DTRACE_SRP_6(name, type1, arg1, type2, arg2, type3, arg3,       \
400             type4, arg4, type5, arg5, type6, arg6)                      \
401         DTRACE_PROBE6(__srp_##name, type1, arg1, type2, arg2,           \
402             type3, arg3, type4, arg4, type5, arg5, type6, arg6);

404 #define DTRACE_SRP_7(name, type1, arg1, type2, arg2, type3, arg3,       \
405             type4, arg4, type5, arg5, type6, arg6, type7, arg7)         \
406         DTRACE_PROBE7(__srp_##name, type1, arg1, type2, arg2,           \
407             type3, arg3, type4, arg4, type5, arg5, type6, arg6, type7, arg7);

409 #define DTRACE_SRP_8(name, type1, arg1, type2, arg2, type3, arg3,       \
410             type4, arg4, type5, arg5, type6, arg6, type7, arg7, type8, arg8) \
411         DTRACE_PROBE8(__srp_##name, type1, arg1, type2, arg2,           \
412             type3, arg3, type4, arg4, type5, arg5, type6, arg6,         \
413             type7, arg7, type8, arg8);

415 /*
416  * The set-error SDT probe is extra static, in that we declare its fake
417  * function literally, rather than with the DTRACE_PROBE1() macro.  This is
418  * necessary so that SET_ERROR() can evaluate to a value, which wouldn't
419  * be possible if it required multiple statements (to declare the function
420  * and then call it).
421  *
422  * SET_ERROR() uses the comma operator so that it can be used without much
423  * additional code.  For example, "return (EINVAL);" becomes
424  * "return (SET_ERROR(EINVAL));".  Note that the argument will be evaluated
425  * twice, so it should not have side effects (e.g. something like:
426  * "return (SET_ERROR(log_error(EINVAL, info)));" would log the error twice).
427  */
428 extern void __dtrace_probe_set__error(uintptr_t);
429 #define SET_ERROR(err) (__dtrace_probe_set__error(err), err)

431 #endif /* _KERNEL */

433 extern const char *sdt_prefix;

435 typedef struct sdt_probedesc {
436         char                    *sdpd_name;     /* name of this probe */
437         unsigned long           sdpd_offset;    /* offset of call in text */
438         struct sdt_probedesc    *sdpd_next;     /* next static probe */
439 } sdt_probedesc_t;
```

_____***unchanged_portion_omitted_***