

Brent O. Hatch (5715)
Mark F. James (5295)
HATCH, JAMES & DODGE, P.C.
10 West Broadway, Suite 400
Salt Lake City, Utah 84101
Telephone: (801) 363-6363
Facsimile: (801) 363-6666

Stephen N. Zack
Mark J. Heise
BOIES, SCHILLER & FLEXNER LLP
100 Southeast Second Street
Suite 2800
Miami, Florida 33131
Telephone: (305) 539-8400
Facsimile: (305) 539-1307

Attorneys for Plaintiff The SCO Group, Inc.

IN THE UNITED STATES DISTRICT COURT
DISTRICT OF UTAH

THE SCO GROUP, INC.
a Delaware corporation,

**PLAINTIFF'S REVISED SUPPLEMENTAL
RESPONSE TO DEFENDANT'S
FIRST AND SECOND SET OF
INTERROGATORIES**

Plaintiff,

vs.

INTERNATIONAL BUSINESS
MACHINES CORPORATION, a
New York corporation,

Honorable Dale A. Kimball
Magistrate Judge Brooke C. Wells

Defendant.

Pursuant to Rule 33 of the Federal Rules of Civil Procedure, and this Court's order dated December 12, 2003, Plaintiff, The SCO Group, Inc. ("SCO"), hereby files this Revised Supplemental Response to Interrogatories No. 1 through 9, 12 and 13.

GENERAL OBJECTIONS

SCO hereby incorporates by reference all of its General Objections set out in Plaintiff's Responses to Defendant's First and Second Set of Interrogatories and First Request for the Production of Documents (the "Plaintiff's Responses"). All of SCO's original General Objections are incorporated into the following Specific Objections and Responses as if fully set forth therein. Pursuant to the Federal Rules of Civil Procedure, SCO's revised and supplemental responses to IBM's Interrogatories are made to the best of SCO's present knowledge, information and belief. In particular, these current responses are based on the evidence SCO has discovered independently and based on information contained in IBM's limited production to date. Upon receiving complete discovery from IBM, including all versions of AIX and Dynix/ptx, there undoubtedly will be further evidence of IBM's contractual breaches and other violations of law. Accordingly, SCO reserves the right to further supplement or amend its answers as discovery or further investigation may reveal.

SPECIFIC OBJECTIONS AND SUPPLEMENTAL RESPONSES TO INTERROGATORIES

INTERROGATORY NO. 1:

Please identify, with specificity (by product, file and line of code, where appropriate) all of the alleged trade secrets and any confidential or proprietary information that plaintiff alleges or contends IBM misappropriated or misused, including but not limited to as alleged in ¶ 105 of the Complaint.

SUPPLEMENTAL RESPONSE TO INTERROGATORY NO. 1:

Subject to and without waiving its objections, Plaintiff supplements its response to this Interrogatory No. 1 and states pursuant to their respective Software Agreements, Sublicensing Agreements and related agreements ("Related Agreements"), which are attached to the Amended Complaint, IBM and Sequent had certain contractual obligations and restrictions on their use of the UNIX System V code that they licensed from AT&T, SCO's predecessor. These restrictions, which are more fully stated in the forgoing agreements, also restricted IBM and Sequent's use of the modifications they made to UNIX System V and derivative works of UNIX System V. IBM's version of UNIX is known as AIX and Sequent's version of UNIX is known as Dynix/ptx. Based on the forgoing agreements, IBM and Sequent agreed to restrictions on AIX and Dynix/ptx, including that AIX and Dynix/ptx would be used solely for internal business purposes, that they would not allow the use of AIX or Dynix/ptx for or by others, and that they would not transfer any part of Dynix/ptx to parties who do not have a UNIX System V source code agreement with SCO. IBM and Sequent also agreed that they would maintain all of AIX and Dynix/ptx in confidence. IBM breached the terms of the agreements and thereby misused or misappropriated the confidential or proprietary or

trade secret information by transferring core portions of AIX and Dynix/ptx to Linux, as detailed below (the "Protected Materials").

Thus far, SCO has received limited production from IBM of some versions of Dynix/ptx for comparison. Based on the limited software produced by IBM, SCO has identified direct copying by IBM of entire files of Dynix/ptx source code as a patch to Linux 2.4.1-01. The first misuse of the Protected Materials identified below is in Read Copy Update ("RCU"). RCU is a mechanism that can significantly improve the performance and scalability of multi-processor systems by allowing simultaneous access to data without the need for expensive and time consuming locking protocols. Dynix/ptx/RCU structures and sequences were originally offered as a patch to the Linux 2.4 kernel by IBM, with rather limited functionality inside Linux 2.4. However, in the development of Linux version 2.6, the deployment of Dynix/ptx/RCU structures and sequences has spread into new uses inside Linux, including networking, device drivers, list management, and directory access. This demonstrates how improper contribution of a few hundred lines of change from Dynix/ptx has had a massive impact on Linux kernel efficiency, particularly relating to multi-processor functionality and processor memory synchronization.

For detailed comparison, compare the files of code contained in Table A below. The original code in Dynix/ptx in Tab 1 is set against the files of code identified in Linux in Tab 5. Compare files contained in Dynix/ptx Tab 2 against the files contained in Linux in Tab 6. Also compare files contained in Tab 3 in Dynix/ptx against files contained in Linux in Tab 7. Compare files contained in Tab 4 in Dynix/ptx against files contained in Tab 8 in Linux. Virtually the entire files identified in the above tabs that originated in Dynix/ptx were published as a patch to Linux 2.4.1-01, with only

minimal changes. As a result, in this particular instance, SCO is only identifying the file names, without additionally specifying the lines of code within each file.

TABLE A

DynixV v4.6.1 Files	Linux 2.4.1-01 files
kernel/sys/rclock.h (Tab 1)	include/linux/rclock.h (Tab 5)
kernel/os/rclock.c (Tab 2)	kernel/rclock.c (Tab 6)
kernel/sys/kma_defer.h (Tab 3)	include/linux/kmemdef.h (Tab 7)
kernel/os/kma_defer.c (Tab 4)	kernel/kmemdef.c (Tab 8)

As stated, the entire files specified above show direct line-by-line copying of the files with the same name in Dynix, with slight changes made to reflect some variations between the two operating systems. By comparing the tabs correlating each file in Dynix against the corresponding patch for Linux, the direct copying is apparent to the layperson's eye. That the code in Linux comes from Dynix/ptx is further confirmed by the commentary in the Linux patch that expressly states that it is "[b]ased on a Dynix/ptx implementation by Paul McKenney..." Mr. McKenney was formerly an engineer at Sequent, and is now employed at IBM following IBM's acquisition of Sequent. After the first initial improper contribution of RCU by IBM, RCU persists as a functionality available in Linux, is maintained by IBM (and thereafter others) and became more widespread in the Linux kernel (as shown in Tab 9).

Lines of code from Dynix/ptx files, but less than the entire file, were also copied line-for-line from DynixV v4.6.1 to Linux 2.4.1-01. Table B maps the line-for-line copied code from specified lines in DynixV v4.6.1 to Linux 2.4.1-01, with the file name and file line number in each code base identified appropriately. By comparing the tabs that correlate with each file in Dynix/ptx against the corresponding file in Linux, the direct copying is again apparent to the untrained eye.

TABLE B

DynixV v4.6.1 Files and line #s		Linux 2.4.1-01 files and line #s	
kernel/os/kern_clock.c (Tab 10)	2028-2059	arch/i386/kernel/apic.c (Tab 14)	25-28, 662-664, 676-684
kernel/os/kern_clock.c (Tab 10)	2028-2059	kernel/timer.c (Tab 15)	26-29, 681-683, 688-697
kernel/i386/locore.s (Tab 11)	1487-1497	arch/i386/kernel/entry.S (Tab 16)	199-205
kernel/i386/trap.c (Tab 12)	1554-1563	arch/i386/kernel/traps.c (Tab 17)	52-54, 244-247, 331-334, 542-545, 659-662, 718-721
kernel/i386/startup.c (Tab 13)	2054	init/main.c (Tab 18)	30-33, 609-616

In Table B, Tab 10 correlates Dynix/ptx code at lines 2028-2059 directly with copies of the same Dynix/ptx code improperly copied into Linux, identified in Tab 14 at lines 25-28, 662-664, 676-684. Tab 10 also correlates the same Dynix/ptx code found at lines 2028-2059 with copies of the same Dynix/ptx code improperly copied into Linux, identified in Tab 15 at lines 26-29, 681-683, 688-697. Tab 11 correlates Dynix/ptx code at lines 1487-1497 directly with copies of the same Dynix/ptx code improperly copied into Linux, identified in Tab 16 at lines 199-205. Tab 12 correlates Dynix/ptx code at lines 1554-1563 directly with copies of the same Dynix/ptx code improperly copied into Linux, identified in Tab 17, lines 52-54, 244-247, 331-334, 542-545, 659-662,

718-721. Tab 13 correlates Dynix/ptx code at line 2054 improperly copied into Linux, identified in Tab 18 at lines 30-33, 609-616. These are instances of direct, line for line copying of Dynix/ptx code into the 2.4.1-01 version of Linux. Prior to this copying, Dynix/ptx had been held in confidence for SCO pursuant to the Sequent Software Agreement.

The next table, Table C, shows an example of UNIX-based code structures and sequences in Dynix/ptx/RCU that have been improperly copied into the newest version of Linux, 2.6.0.

TABLE C

Structure / Sequence	DynixV file(s)	Line #s	Linux 2.6.0 file(s)	Line #s
Register an RCU callback	kernel/sys/rclock.h (Tab 1)	412	include/linux/rcupdate.h (Tab 20)	131-132
	kernel/os/rclock.c (Tab 2)	503-613	kernel/rcupdate.c (Tab 21)	58-80
RCU Callback control structure	kernel/sys/rclock.h (Tab 1)	217-228	include/linux/rcupdate.h (Tab 20)	66-72
RCU Callback lists	kernel/sys/rclock.h (Tab 1)	238-241	include/linux/rcupdate.h (Tab 20)	87-108
	kernel/i386/plocal.h (Tab 19)	1517-1537		
RCU Comparison operators	kernel/sys/rclock.h (Tab 1)	300-312	include/linux/rcupdate.h (Tab 20)	75-85

The sequence that performs "Register an RCU callback" in DynixV v4.6.1 at line 412, (Tab 1) correlates to the same sequence improperly copied into, and used in Linux 2.6.0 at line 131-132 (Tab 20). The structure "RCU callback control" performed in Dynix/ptx at line 127-228 (Tab 2) was improperly copied into Linux 2.6.0 at lines 66-72 (Tab 21). The sequence "RCU callback lists" performed in Dynix/ptx at line 238-241 (Tab 1) and lines 1517-1537 (Tab 19) was improperly copied into Linux 2.6.0 at lines 87-108 (Tab 20). The structure "RCU comparison operators" found in Dynix/ptx at line 300-312 (Tab 1) was improperly copied into Linux 2.6.0 at lines 75-85 (Tab 20). Again, these are structures and sequences that materially advance the enterprise performance of Linux, and are based on the Protected Materials that Sequent had agreed to not disclose.

The next table, Table D shows how IBM has released valuable proprietary methods related to RCU functionality found in Dynix/ptx that have now been adapted across the newest release of Linux, version 2.6.0 in a multitude of different ways. A sub-component of RCU is “force cache write for memory consistency” found in Dynix/ptx at lines 331-358.

TABLE D

RCU Subcomponent	DynixV file(s)	Line #s	Linux 2.6.0 file(s)	Line #s
Force Cache write for memory consistency	kernel/sys/rclock.h (Tab 1)	331-358	include/asm-alpha/system.h (Tab 22)	152-153, 159, 164
			include/asm-arm/system.h (Tab 23)	104, 228, 235
			include/asm-arm26/system.h (Tab 24)	197
			include/asm-cris/system.h (Tab 25)	19, 27, 32
			include/asm-h8300/system.h (Tab 26)	97, 102
			include/asm-i386/system.h (Tab 27)	368-420, 434, 440
			include/asm-ia64/system.h (Tab 28)	83, 89, 94
			include/asm-m68k/system.h (Tab 29)	80, 87
			include/asm-m68knommu/system.h (Tab 30)	105, 110
			include/asm-mips/system.h (Tab 31)	145-197, 256, 261
			include/asm-parisc/system.h (Tab 32)	134-135
			include/asm-ppc/system.h (Tab 33)	24-25, 34, 43, 48
			include/asm-ppc64/system.h (Tab 34)	28-29, 38, 47, 52
			include/asm-s390/system.h (Tab 35)	255, 259
			include/asm-sh/system.h (Tab 36)	80, 86, 91
			include/asm-sparc/system.h (Tab 37)	281, 287
			include/asm-sparc64/system.h (Tab 38)	86, 96, 101
			include/asm-v850/system.h (Tab 39)	70, 78
			include/asm-x86_64/system.h (Tab 40)	283, 288, 305

Thus, Dynix/ptx lines 331-358 maps in structure and sequence to Linux 2.6.0 files shown in Table D. These include lines of the Linux 2.6.0 kernel as follows: lines 152-153, 159, 164 (Tab 22); lines 104, 228, 235 (Tab 23); line 197 (Tab 24); lines 19, 27, 32 (Tab 25); lines 97-102 (Tab 26); lines 368-420, 434, 440 (Tab 27); lines 83, 89, 94 (Tab 28); lines 80,87 (Tab 29); lines 105, 110 (Tab 30); lines 145-197, 256, 261 (Tab 31); lines 134-135 (Tab 32); lines 24-25, 34, 43, 48 (Tab 33); 28-29, 38, 47, 52 (Tab 34); lines 255, 259 (Tab 35); lines 80, 86, 91 (Tab 36); lines 281, 287 (Tab 37); lines 86, 96, 101 (Tab 38); lines 70, 78 (Tab 39); lines 283, 288, 305 (Tab 40). This is a good demonstration of how the Protected Materials that were subject to the restrictions in the Software Agreements, Sublicensing Agreements and Related Agreements have rapidly spread throughout Linux by the open source development community, and underscores why IBM's improper contributions to Linux are so devastating to the value of proprietary UNIX technology. IBM is literally giving Protected Materials to the open source development community so that Linux can be made suitable for enterprise use. But for IBM's wrongful actions in this regard, Linux would lag far behind SCO's UnixWare in functionality and acceptability for enterprise use on Intel processors.

As is demonstrated above, and in the following examples as well, IBM's Linux plan has been to release large blocks of proprietary source code and methods to open source, and to invite the entire open source development community to access that code and those methods in their latest design efforts for Linux. IBM has provided the valuable UNIX proprietary code and methods, and the Protected Materials, to the open source development community. Open source developers have then accessed the Protected Materials at will to design new creations in Linux based on the Protected Materials. Another example of this problem is shown in the next table, Table E. Table E demonstrates how use of the RCU method, which is subject to the restrictions of the Sequent

agreements and is prohibited from improper disclosure, has been improperly used in Linux far beyond its original use in Dynix/ptx.

TABLE E

RCU Method	DynixV file(s)	Line #s	Linux 2.6.0 file(s)	Line #s
RCU use	<p>After IBM's initial improper contribution of RCU specifically identified above, RCU became widespread in the Linux kernel</p> <p>These files use RCU functions and macros – and if they are used, then the RCU functionality is used, too. The number of lines and number of files, although substantial, is less significant than the impact of the changes, which is dramatic for networking, device drivers, lists, and directory access.</p>		net/core/netfilter.c (Tab 41)	73, 83, 357, 516-517, 548, 558, 575, 612
			net/core/dev.c (Tab 42)	239, 242, 238, 995-996, 1027, 1580-1581, 1594, 1614, 2893
			net/ipv4/ip_input.c (Tab 43)	220, 241, 266
			net/ipv4/af_inet.c (Tab 44)	340-341, 375, 430, 1012, 1040
			net/ipv4/icmp.c (Tab 45)	707-712
			net/ipv4/route.c (Tab 46)	227, 231, 240, 243, 246, 282, 440, 446, 1004, 1008, 1026, 1037, 1076, 1257, 1260, 1296, 1852, 1854, 1867, 1873, 2219, 2221, 2235, 2241, 2451, 2454, 2462, 2467
			net/bridge/br_device.c (Tab 47)	79, 81
			net/bridge/br_ioctl.c (Tab 48)	78, 97, 159, 161, 179
			net/bridge/br_stp.c (Tab 49)	43
			net/bridge/br_private.h (Tab 50)	77
			net/bridge/br_input.c (Tab 51)	59, 61, 106, 117, 135, 142, 151, 156

	net/bridge/br_forward.c (Tab 52)	77-97, 120, 147, 151
	net/bridge/br_if.c (Tab 53)	64, 72, 160, 269-270, 273
	net/irda/irlan/irlan_common.c (Tab 54)	230, 275, 1083, 1113
	net/irda/irlan/irlan_client.c (Tab 55)	172, 182
	net/ipv6/icmp.c (Tab 56)	532, 534, 537
	net/ipv6/af_inet6.c (Tab 57)	173-174, 208, 271, 275, 279, 614, 640
	net/ipv6/ip6_input.c (Tab 58)	155, 169, 197, 201
	net/802/psnap.c (Tab 59)	36, 58, 71, 136, 151
	net/decnet/dn_route.c (Tab 60)	149, 156, 1171, 1173, 1184, 1189, 1450, 1452, 1463, 1468, 1644, 1646, 1653, 1658, 1678, 1682, 1691, 1694, 1697, 1723
	drivers/net/wireless/strip.c (Tab 61)	973, 983, 998, 1006, 2562
	drivers/net/wan/lapbether.c (Tab 62)	70, 98, 118, 379, 395, 411, 431
	drivers/net/hamradio/bpqether.c (Tab 63)	183, 225, 408, 437, 545, 560, 575, 595
	drivers/char/ipmi/ipmi_kcs_intf.c (Tab 64)	1203
	arch/i386/oprofile/nmi_timer_int.c (Tab 65)	41
	include/linux/dcache.h (Tab 66)	96, 180
	include/linux/list.h (Tab 67)	83-124, 152-167, 367-414, 462-477, 487, 499-508
	include/net/dst.h (Tab 68)	77

		fs/dcache.c (Tab 69)	81, 986, 993, 1015, 1038, 1146, 1232, 1235
		kernel/module.c (Tab 70)	1738
		ipc/util.c (Tab 71)	197, 282, 289, 349, 354, 459, 461, 475, 478, 488, 497
		init/main.c (Tab 72)	414

The next table, Table F, shows additional misuse by IBM of the RCU structures and sequences embodied in Dynix/ptx.

TABLE F

RCU sub-component	DynixV file(s)	Line #s	Linux 2.6.0 file(s)	Line #s
RCU read protect	kernel/sys/rclock.h (Tab 1)	373-387	include/linux/rcupdate.h (Tab 20)	124-125
	kernel/os/rclock.c (Tab 2)	1758-1825		
	kernel/os/rclock.c (Tab 2)	503-613	kernel/rcupdate.c (Tab 21)	58-80
Existence of valid callbacks, call "checker"	kernel/os/kern_clock.c (Tab 10)	2028-2059	include/linux/rcupdate.h (Tab 20)	112-122
			kernel/sched.c (Tab 73)	1364-1365
RCU "checker" (actually processes callbacks)	kernel/sys/rclock.h (Tab 1)	411, 415	include/linux/rcupdate.h (Tab 20)	128
	kernel/os/rclock.c (Tab 2)	385-468, 659-752, 1314-1494	kernel/rcupdate.c (Tab 21)	82-207
RCU initialization	kernel/sys/rclock.h (Tab 1)	414	include/linux/rcupdate.h (Tab 20)	127
	kernel/os/rclock.c (Tab 2)	1222-1311	kernel/rcupdate.c (Tab 21)	201-240

The RCU subcomponent identified as "RCU read protect" is found in Dynix/ptx at lines 373-387 (Tab 1) and lines 1758-1825 (Tab 2). These have been improperly copied into Linux 2.6.0 at lines 124-125 (Tab 20). The RCU subcomponent identified as "Existence of valid callbacks, call checker" is found in Dynix/ptx at lines 2028-2059 (Tab 10). These have been improperly copied into Linux 2.6.0 at lines 112-133 (Tab 20) and 1364-1365 (Tab 73). The RCU subcomponent known as "RCU checker (actually processes callbacks)" is found in Dynix/ptx at lines 411, 415 (Tab 1); lines

385-468, 659-752, 1314-1494 (Tab 2). These have been improperly copied into Linux 2.6.0 at lines 82-207 (Tab 21). The RCU subcomponent known as “RCU initialization” is found in Dynix/ptx at lines 414 (Tab 1) and lines 1222-1311 (Tab 2). These have been improperly copied into Linux 2.6.0 at lines 127 (Tab 20) and 201-240 (Tab 21).

An additional core technology transferred improperly by IBM to Linux from Dynix/ptx and AIX is in asynchronous input/output (“AIO”) and scatter/gather I/O. Input/output (“I/O”) is the way operating systems communicate with files and peripheral devices attached to the computer (such as a standard printer or a network interface). Asynchronous and scatter/gather I/O are specialized methods for I/O that have recently been included into Linux and, as discussed in detail below, are believed to have originated in Dynix/ptx and/or AIX and therefore are part of the Protected Materials.

The Linux “patches” implementing AIO were provided by Badari Pulavarty, formerly a Sequent employee, now an IBM employee. Improper transfer of AIO Protected Materials to Linux is illustrated below in this comment from Linux 2.6.0, in the file fs/direct-io.c:

```
/*
 * fs/direct-io.c
 *
 * Copyright (C) 2002, Linus Torvalds.
 *
 * O_DIRECT
 *
 * 04Jul2002  akpm@zip.com.au
 *           Initial version
 * 11Sep2002  janetinc@us.ibm.com
 *           added readv/writev support.
 * 29Oct2002  akpm@zip.com.au
 *           rewrote bio_add_page() support.
 * 30Oct2002  pbadari@us.ibm.com
 *           added support for non-aligned IO.
 * 06Nov2002  pbadari@us.ibm.com
 *           added asynchronous IO support.
```

* 21Jul2003 nathans@sgi.com
 * added IO completion notifier.
 */

(Emphasis added).

In October and November of 2002, Badari Pulavarty, formerly a Sequent employee, now an IBM employee, added changes that allow for asynchronous I/O and non-aligned I/O to Linux. In order to completely trace the Dynix/ptx transfer of AIO technology to Linux, SCO needs to obtain the full production of source code from IBM. However, from examining v4.6.1 of Dynix/ptx, it is apparent that Badari Pulavarty was an experienced Dynix kernel programmer, based on the numerous revisions to Dynix with which he is credited, including the following:

Dynix v4.6.1 file	Revisions credited to Badari
./io/tlbtest/tlbtest rc.c	Revision 1.28
./io/tlbtest/tlbtest tout.c	Revision 1.18
./io/ff/ff fc.c	Revision 1.111, Revision 1.110
./kernel/debug/dinfo.c	Revision 1.75
./kernel/i386/mc_vmmac.h	Revision 1.53
./kernel/i386/plocal.h	Revision 1.144, Revision 1.143
./kernel/i386/vm_boot.c	Revision 1.199
./kernel/i386/vmpt_machdep.c	Revision 1.180, Revision 1.101
./kernel/i386_space/param space.c	Revision 1.207
./kernel/os/heap_kmem.c	Revision 1.128
./kernel/os/init_main.c	Revision 1.216
./kernel/os/kern_fork.c	Revision 1.253, Revision 1.250, Revision 1.245, Revision 1.238
./kernel/os/kern_exec.c	Revision 1.200, Revision 1.198, Revision 1.197, Revision 1.196
./kernel/os/kern_sig.c	PRs 254728, 254503, SCN sarahw186, Reviewer: badari
./kernel/os/mmap_ifchr.c	Revision 1.55
./kernel/os/kern_posix.c	Revision 1.43
./kernel/os/sys_process.c	Revision 1.267
./kernel/os/vm_sched.c	Revision 1.145, Revision 1.143, Revision 1.130, Revision 1.122
./kernel/os/vm_sched.c	Revision 1.121
./kernel/os/sys_vm.c	Revision 1.57
./kernel/os/vfs_bio.c	Revision 1.108, Revision 1.106, Revision 1.105
./kernel/os/kern_clock.c	Revision 1.168, Revision 1.165
./kernel/os/vm_swap.c	Revision 1.163, Revision 1.156, Revision 1.153
./kernel/os/vm_drum.c	Revision 1.141, Revision 1.138

./kernel/os/vm_mem.c	Revision 1.216, Revision 1.215, Revision 1.211, Revision 1.210, Revision 1.209, Revision 1.206, Revision 1.196, Revision 1.195, Revision 1.194, Revision 1.192
./kernel/os/vm_page.c	Revision 1.126, Revision 1.125
./kernel/os/vm_page.c	Revision 1.122
./kernel/os/vm_pageout.c	Revision 1.100, Revision 1.97
./kernel/os/vm_proc.c	Revision 1.118, Revision 1.117
./kernel/os/vm_sw.c	Revision 1.140
./kernel/os/vm_subr.c	Revision 1.88, Revision 1.86
./kernel/os/vm_swap.c	Revision 1.151
./kernel/os/mmap_mfile.c	Revision 1.148, Revision 1.147, Revision 1.145, Revision 1.142
./kernel/os/mmap_ifreg.c	Revision 1.146, Revision 1.144, Revision 1.143, Revision 1.142, Revision 1.140, Revision 1.138, Revision 1.136
./kernel/os/audit_subr.c	Reviewers: dmo, badari
./kernel/os/mmap_anon.c	Revision 1.121, Revision 1.119, Revision 1.114, Revision 1.106
./kernel/os/vm_asops.c	Revision 1.210, Revision 1.203, Revision 1.196
./kernel/os/vfs_dio.c	Revision 1.86, Revision 1.85
./kernel/os/kern_perf.c	Revision 1.63
./kernel/os/kern_daemon.c	Revision 1.43
./kernel/os/kern_lwp.c	Revision 1.95, Revision 1.92, Revision 1.88, Revision 1.85
./kernel/os/kern_lwkdir.c	PR #254650, SCN sarahw163, reviewer : badari
./kernel/os/lrwsema.c	Revision 1.17
./kernel/os/region.c	Revision 1.48, Revision 1.47, Revision 1.44, Revision 1.41, Revision 1.39
./kernel/os/region_mem.c	Revision 1.57, Revision 1.56, Revision 1.53, Revision 1.51, Revision 1.49, Revision 1.48, Revision 1.45, Revision 1.44, Revision 1.42, Revision 1.37, Revision 1.35, Revision 1.34, Revision 1.32
./kernel/os/region_misc.c	Revision 1.19, Revision 1.18
./kernel/sys/region.h	Revision 1.65, Revision 1.64, Revision 1.60, Revision 1.58, Revision 1.57
./kernel/sys/swap.h	Revision 1.35
./kernel/sys/region_kstats.h	Revision 1.8
./kernel/sys/mman.h	Revision 1.111, Revision 1.108
./kernel/sys/aumacros.h	PR 238431; SCN gerrit716; Reviewers: dmo, badari
./kernel/sys/ucontext.h	PR 254872, SCN sarahw186, Reviewer badari
./kernel/sys/buf.h	Revision 1.61
./kernel/sys/vm_extern.h	Revision 1.121, Revision 1.120, Revision 1.117
./kernel/sys/cmap.h	Revision 1.61, Revision 1.60, Revision 1.58, Revision 1.56, Revision 1.55
./kernel/sys/autotypes.h	PR 251279; SCN timw369; Reviewer: badari
./kernel/sys/region_misc.h	Revision 1.7
./kernel/sys/proc.h	Revision 1.214

./kernel/sys/region_mem.h	Revision 1.27, Revision 1.21, Revision 1.20, Revision 1.18
./kernel/vm/vmdki.c	Revision 1.179
./kernel/vm/vm_ublock.c	Revision 1.37
./kernel/sci/sci_archdep.c	reviewer - badari
./kernel/proc/migrate.c	Revision 1.146, Revision 1.145, Revision 1.136
./kernel/scheduler/sched_core.c	Revision 1.265
./kernel/scheduler/sched_loadbal.c	Revision 1.3
./kernel/scheduler/sched_runq.c	Revision 1.5
./shm/shm.c	Revision 1.123
./ufs/ufs_inode.c	Revision 1.117
./ufs/ufs_vnodeops.c	PR #254506, SCN sarahw189, reviewer : badari

The scope and type of changes indicate that Badari is a programmer who is intimately familiar with the UNIX kernel, and has considerable experience with the Dynix kernel. More specifically, in Dynix/ptx v4.6.1 a file named kernel/os/vfs_dio.c has the following comments attributed to Badari:

- * Revision 1.86 1999/06/16 23:17:48 badari
- * Revision 1.85 1999/05/03 23:41:53 badari

These comments are specific to fixing aspects of the asynchronous I/O implementation. More significantly, the file vfs_dio.c in Dynix implements the "direct I/O" subsystem, which is coupled to implementation of asynchronous I/O and scatter/gather. The file kernel/os/vfs_dio.c in Dynix/ptx ("Virtual File System/Direct I/O") has direct relation to the file fs/direct_io.c in Linux 2.6.0. Based on the forgoing, SCO has reason to believe that through Badari's contributions to Linux, IBM improperly transferred Protected Materials to Linux. Badari Pulavarti certainly was in a position to contribute expertise (and apparently implementation) to the Linux kernel that most likely was gained while working on Dynix/ptx. This belief is based, in part, on the observation that most of Badari's contributions to Dynix are 1999/2000 while his contributions to Linux are dated 2002. Further comments in newsgroups and Linux patches support our belief. To definitively trace the Dynix/ptx transfer of AIO technology to Linux, SCO needs to obtain the full production of source code from

IBM. In summary, key components of UNIX high-performance systems appear (based on Badari's own comments) to have been contributed to Linux by a programmer who had worked on the same subsystem of Dynix/ptx 2 years earlier.

Further, improper transfer of scatter/gather I/O Protected Materials to Linux is illustrated in the same comments in the same Linux 2.6.0 file (`fs/direct_io.c`). The scatter/gather mechanism "readv" and "writev" was contributed by Janet Morgan from IBM (janetinc@us.ibm.com). Ms. Morgan has been identified by IBM as someone who has or had access to AIX source code. Through Janet Morgan's contributions to readv/writev in Linux, IBM improperly transferred Protected Materials to Linux. Further comments in newsgroup discussions support our belief. In order to completely trace the AIX transfer of scatter/gather technology to Linux, SCO needs to obtain the full production of source code from IBM. In summary, another key component of UNIX high-performance systems appears (based on Ms. Morgan's own comments) to have been contributed to Linux by a programmer who had access to the same subsystem of AIX.

IBM has also improperly contributed other core technologies found in IBM's own derivative work of UNIX known as AIX to Linux in violation of the IBM Related Agreements. As noted earlier, IBM agreed in the Software Agreement that it would use AIX solely for internal business purposes, that it would not allow the use of AIX for or by others, and that it would not transfer any part of AIX to parties who do not have a UNIX System V source code agreement with SCO. IBM also agreed that it would maintain all of AIX in confidence, and that it would not adapt or allow its contactors to adapt AIX for purposes of a creation of a new general operating system by a non-IBM entity. IBM breached its promises to SCO in the Software Agreement, Sublicensing Agreement and Related Agreements by transferring core portions of AIX to Linux.

Thus far, SCO has received no production whatsoever from IBM of AIX software. Notwithstanding this fact, SCO has identified copying by IBM of files of AIX into Linux. One instance of copying of AIX into Linux involves improper contribution by IBM to Linux 2.4 of the AIX Journaling File System ("JFS"). The contribution of JFS was done in a series of "drops" of AIX code identified as "reference files" inside Linux. The first such drop occurred on or about February 2000, with multiple additions and significant follow-up work by IBM since that time to adapt AIX/JFS for enterprise use inside Linux. These drops of reference files do not necessarily become part of the source code in the Linux kernel, but rather are public displays of the Protected Materials so that anyone has access to them and can use them to construct a similar file in Linux.

The first drop contains (a) partially functioning port, or transfer, of JFS from AIX to Linux; (b) a set of reference directories (named ref/) which contain the AIX reference version of AIX/JFS; (c) AIX/JFS-related utility files used to maintain and upkeep AIX/JFS; and (d) a set of directories (named directory ref_utils/) which contain the AIX reference version of utilities. Copies of AIX/JFS files into Linux are shown in Table G, below. Table G compares a 1999 version of AIX currently in SCO's possession. Nevertheless, even this old version of AIX shows the following similarities, demonstrating copying of code, structures and sequences.

TABLE G

AIX 9922A 43NIA File	Line #s	Linux 2.2.12 ref/ File	Line #s
usr/include/jfs/inode.h (Tab 74)	16-37	include/linux/jfs/ref/jfs inode.h (Tab 76)	84-95, 126-138
kernel/sys/vnode.h (Tab 75)	109-133	include/linux/jfs/ref/jfs inode.h (Tab 76)	96-122
usr/include/jfs/inode.h (Tab 74)	39-40	include/linux/jfs/ref/jfs inode.h (Tab 76)	189-90
usr/include/jfs/inode.h (Tab 74)	161-166	include/linux/jfs/ref/jfs inode.h (Tab 76)	414-421
usr/include/jfs/inode.h (Tab 74)	172-180	include/linux/jfs/ref/jfs inode.h (Tab 76)	37-48
usr/include/jfs/inode.h (Tab 74)	199-205	include/linux/jfs/ref/jfs inode.h (Tab 76)	52-59
usr/include/jfs/inode.h (Tab 74)	62-66	include/linux/jfs/ref/jfs inode.h (Tab 76)	286-290
usr/include/jfs/inode.h (Tab 74)	72-76	include/linux/jfs/ref/jfs inode.h (Tab 76)	295-302
usr/include/jfs/inode.h (Tab 74)	83-158	include/linux/jfs/ref/jfs inode.h (Tab 76)	322-411

Protected Materials from AIX appear in AIX 9922A_43NIA (hereafter referred to only as "AIX") at lines 16-37 (Tab 74) and have been improperly copied into Linux version 2.2.12 at lines 84-95, 126-138 (Tab 75). Protected Materials from AIX at lines 109-133 (Tab 75) have been improperly copied into Linux at lines 96-122 (Tab 76). Protected Materials from AIX at lines 39-40 (Tab 74) have been improperly copied into Linux at lines 189-90 (Tab 76). Protected Materials from AIX at lines 161-166 (Tab 74) have been improperly copied into Linux at lines 414-421 (Tab 76). Protected Materials from AIX at lines 172-180 (Tab 74) have been improperly copied to Linux at lines 37-48 (Tab 76). Protected Materials from AIX at lines 199-205 (Tab 74) have been improperly copied to Linux at lines 52-59 (Tab 76). Protected Materials from AIX at lines 62-66 (Tab 74) have been improperly copied to Linux at lines 286-290 (Tab 76). Protected Materials from AIX at lines 72-76 (Tab 74) have been improperly copied to Linux at lines 295-302 (Tab 76). Protected Materials from AIX at lines 83-158 (Tab 74) have been copied improperly to Linux at lines 322-411 (Tab 76). These transfers of AIX/JFS to Linux are in violation of the Related Agreements, and are an improper use of AIX for adaptation to a general operating system in violation of the Related Agreements.

In addition, there are source code files of Protected Materials in JFS called "aixisms.h" which demonstrate the AIX core nature of JFS. Linux files referencing AIX (by name in the commentary) in the JFS drops identified in the AIX files and lines include those listed in Table H.

TABLE H

Files	Lines
include/linux/jfs/ref/jfs_aixisms.h (Tab 77)	26-27, 32, 62, 193, 227, 248
include/linux/jfs/ref/jfs_dirent.h (Tab 78)	55
include/linux/jfs/ref/jfs_inode.h (Tab 76)	76-77, 81, 95, 97
include/linux/jfs/ref/jfs_os2.h (Tab 79)	33-34

fs/jfs/ref/jfs_dio.c (Tab 80)	333
fs/jfs/ref/jfs_logmgr.c (Tab 81)	3134

These files were improperly transferred by IBM to Linux as part of IBM's effort to transfer JFS capabilities from AIX to Linux. The files that include "AIXisms," improperly transferred by IBM, are included in Linux at lines 26-27, 32, 62, 193, 227, 248 (Tab 77); line 55 (Tab 78); lines 76-77, 81, 95, 97 (Tab 75); line 33-34 (Tab 79); line 333 (Tab 80) and line 3134 (Tab 81). At this time, there is clear proof of IBM's use of AIX in contributing JFS to Linux in violation of its contractual obligations. The purpose of making these files available in Linux was to assist the open source development community to use AIX/JFS to improve or perfect a Journaling File System for Linux. By including the "AIXisms", developers are able to see the original source code, see how JFS worked in AIX, modify Linux code based on IBM's significant experience with JFS in AIX in enterprise applications, and thereby gain the benefit of IBM's nearly 20 years of UNIX programming in adapting AIX/JFS for Linux. In Tab 80, for example, in the file marked *fs/jfs/ref/jfs_dio.c* (Tab 80) in the table above, the entry in Linux, written by IBM, says "[o]n AIX we were able to do this in dioIODone () function." This represents a clear violation of IBM's obligations to not transfer AIX to contractors for adaptation for a general operating system.

IBM's decision to release AIX/JFS source code into reference files in Linux illustrates the early pattern of IBM's Linux development plan. Under that plan, IBM initially placed AIX/JFS into files that (mostly) did not actually operate within Linux at the time of the source code drop. Rather, the AIX/JFS files were simply placed in a separate, non-compiling, file that allowed open source programmers to see and use UNIX/AIX development methods and code to improve Linux. JFS then, became another source of UNIX protected technology transferred by IBM to assist in the growth of

enterprise Linux. In addition, IBM's drop of JFS into Linux reference files contains references to the UNIX-based header files, not otherwise found in Linux prior to IBM's identified transfers, further indicating that the source of this technology was AIX.

These reference files are listed in Table I below.

TABLE I

AIX JFS Reference File in Linux	Header file used
Include/linux/jfs/ref/jfs_dasdim.h (Tab 82)	<net32/netcons.h>
	<net32/neterr.h>
	<net32/dasd.h>
Include/linux/jfs/ref/jfs_dinode.h (Tab 83)	<sys/types.h>
	<sys/mode.h>
	<sys/time.h>
	<sys/lock_def.h>
Include/linux/jfs/ref/jfs_lock.h (Tab 84)	"mmph.h"
include/linux/jfs/ref/jfs_superblock.h (Tab 85)	<sys/time.h>
fs/jfs/ref/jfs_bufmgr.c (Tab 86)	"mmph.h"
fs/jfs/ref/jfs_cachemgr.c (Tab 87)	"mmph.h"
fs/jfs/ref/jfs_dio.c (Tab 88)	"mmph.h"
fs/jfs/ref/jfs_dnlc.c (Tab 89)	"mmph.h"
fs/jfs/ref/jfs_dtree.c (Tab 90)	"mmph.h"
fs/jfs/ref/jfs_ifs.c (Tab 91)	"mmph.h"
fs/jfs/ref/jfs_initl.c (Tab 92)	<devcmd.h>
fs/jfs/ref/jfs_inode.c (Tab 93)	"mmph.h"
fs/jfs/ref/jfs_link.c (Tab 94)	<sys/vnode.h>
	<sys/errno.h>
fs/jfs/ref/jfs_logmgr.c (Tab 95)	"mmph.h"
fs/jfs/ref/jfs_mknod.c (Tab 96)	<sys/vfs.h>
	<sys/cred.h>
	<sys/errno.h>
fs/jfs/ref/jfs_readdir.c (Tab 97)	"mmph.h"
fs/jfs/ref/jfs_readlink.c (Tab 98)	<sys/file.h>
	<sys/errno.h>
fs/jfs/ref/jfs_statfs.c (Tab 99)	<sys/vfs.h>
	<sys/statfs.h>
fs/jfs/ref/jfs_symlink.c (Tab 100)	<sys/vfs.h>
	<sys/uio.h>

	<sys/file.h>
	<sys/cred.h>
	<sys/errno.h>
fs/jfs/ref/jfs_txnmgr.c (Tab 101)	"mmph.h"
fs/jfs/ref/selector.c (Tab 102)	<seldesc.h>

The use of these header files provides exact sequences and direct code for use by Linux programmers to copy into Linux when building a new journaling file system for Linux, which in this case is clearly derived from, and based on, AIX/JFS. On information and belief, much of the C source code contained in the files referenced by the above header files was also transferred improperly by IBM to Linux. Without the later versions of AIX it is not possible to definitively make this statement, but SCO expects to confirm this fact upon receipt of outstanding discovery requests from IBM, including recent versions of AIX. In numerous files in the Linux JFS directory (that is, the directory of files that actually compile into Linux, as opposed to the reference files which will not compile under Linux), there are indications that the AIX source code has now been used by the open source development community as a template for creation of the new journaling file system for Linux improperly based on AIX/JFS. In addition to code similarities, there are also compiler directives that serve as comments. For example, the code in the Linux file entitled "fs/jfs/jfs_umount.c" at lines 35-55 (Tab 110) states as follows:

```

35 #include <linux/fs.h>
36 #include <linux/jfs/jfs_types.h>
37 #include <linux/jfs/jfs_filsys.h>
38 #include <linux/jfs/jfs_superblock.h>
39 #include <linux/jfs/jfs_imap.h>
40 #include <linux/jfs/jfs_debug.h>

42 #ifdef _STILL_TO_PORT
43 #include "jfs_types.h"
44 #include "jfs_filsys.h"

```

```

45 #include "jfs_lock.h"
46 #include "jfs_inode.h"
47 #include "jfs_bufmgr.h"
48 #include "jfs_superblock.h"
49 #include "jfs_imap.h"
50 #include "jfs_dmap.h"
51 #include "jfs_dnln.h"
52 #include "jfs_proto.h"
53 #include "jfs_dasdim.h"
54 #include "jfs_debug.h"
55 #endif /* _STILL_TO_PORT */

```

Lines 35-40 above use the header files specified and now actually compile and run Linux code based on AIX/JFS. However, the next line, line 42, includes a compiler directive that also serves as comments that instruct the programmer as follows:

“If the symbol `_STILL_TO_PORT` is defined, then use the header files until you see `#endif /* _STILL_TO_PORT */` (on line 55).

This comment tells the programmer how to work around AIX reference files that still will not compile in Linux, while allowing the ones that now function with Linux to compile, and thereby operate as the new Linux filing system based on AIX/JFS. This is the equivalent of telling a programmer that “work here is yet to be done”, while at the same time providing a partially working system. The commentary is not limited to inclusion of header files, but also includes actual code. Wherever these comments are present, the non-ported code is identical to that in the AIX reference files, and the ported code bears striking similarities to the original. Other places in Linux where this partial-port commentary exists are set forth in Table J.

TABLE J

Linux 2.2.12 File	Line #s
include/linux/jfs/jfs_dmap.h (Tab 104)	31-36, 291-329, 158-165