



*Aegis
Command
Reference*

002547-A00

apollo

Aegis Command Reference

Order No. 002547-A00

Apollo Computer Inc.
330 Billerica Road
Chelmsford, MA 01824

Confidential and Proprietary. Copyright © 1988 Apollo Computer, Inc., Chelmsford, Massachusetts.
Unpublished — rights reserved under the Copyright Laws of the United States. All Rights Reserved.

First Printing: July 1988

Apollo and Domain are registered trademarks of Apollo Computer Inc.

UNIX is a registered trademark of AT&T in the USA and other countries.

3DGMR, Aegis, D3M, DGR, Domain/Access, Domain/Ada, Domain/Bridge, Domain/C, Domain/ComController, Domain/CommonLISP, Domain/CORE, Domain/Debug, Domain/DFL, Domain/Dialogue, Domain/DQC, Domain/IX, Domain/Laser-26, Domain/LISP, Domain/PAK, Domain/PCC, Domain/PCI, Domain/SNA, Domain X.25, DPSS, DPSS/Mail, DSEE, FPX, GMR, GPR, GSR, NLS, Network Computing Kernel, Network Computing System, Network License Server, Open Dialogue, Open Network Toolkit, Open System Toolkit, Personal Supercomputer, Personal Super Workstation, Personal Workstation, Series 3000, Series 4000, Series 10000, and VCD-8 are trademarks of Apollo Computer Inc.

Apollo Computer Inc. reserves the right to make changes in specifications and other information contained in this publication without prior notice, and the reader should in all cases consult Apollo Computer Inc. to determine whether any such changes have been made.

THE TERMS AND CONDITIONS GOVERNING THE SALE OF APOLLO COMPUTER INC. HARDWARE PRODUCTS AND THE LICENSING OF APOLLO COMPUTER INC. SOFTWARE PROGRAMS CONSIST SOLELY OF THOSE SET FORTH IN THE WRITTEN CONTRACTS BETWEEN APOLLO COMPUTER INC. AND ITS CUSTOMERS. NO REPRESENTATION OR OTHER AFFIRMATION OF FACT CONTAINED IN THIS PUBLICATION, INCLUDING BUT NOT LIMITED TO STATEMENTS REGARDING CAPACITY, RESPONSE-TIME PERFORMANCE, SUITABILITY FOR USE OR PERFORMANCE OF PRODUCTS DESCRIBED HEREIN SHALL BE DEEMED TO BE A WARRANTY BY APOLLO COMPUTER INC. FOR ANY PURPOSE, OR GIVE RISE TO ANY LIABILITY BY APOLLO COMPUTER INC. WHATSOEVER.

IN NO EVENT SHALL APOLLO COMPUTER INC. BE LIABLE FOR ANY INCIDENTAL, INDIRECT, SPECIAL OR CONSEQUENTIAL DAMAGES WHATSOEVER (INCLUDING BUT NOT LIMITED TO LOST PROFITS) ARISING OUT OF OR RELATING TO THIS PUBLICATION OR THE INFORMATION CONTAINED IN IT, EVEN IF APOLLO COMPUTER INC. HAS BEEN ADVISED, KNEW OR SHOULD HAVE KNOWN OF THE POSSIBILITY OF SUCH DAMAGES.

THE SOFTWARE PROGRAMS DESCRIBED IN THIS DOCUMENT ARE CONFIDENTIAL INFORMATION AND PROPRIETARY PRODUCTS OF APOLLO COMPUTER INC. OR ITS LICENSORS.

Preface

The *Aegis Command Reference* provides complete reference information on all the Aegis™ commands that are available to you. We assume that you are already familiar with the material in *Getting Started with Domain/OS*. Basics like file structure and usage are taken for granted here: this manual tells you how to use commands, not why you might want to use them.

We've divided the manual into two parts. Chapter 1 summarizes the basic concepts that apply to the Aegis commands; Chapter 2 describes each command individually.

Documentation Conventions

This manual uses the following symbolic conventions:

commands and keywords	Bold words or characters in formats and command descriptions represent commands or keywords that you must use literally. Bold words in text indicate the first use of a new term. Filenames and pathnames are also in bold.
<i>user-supplied values</i>	Italic words or characters in formats and command descriptions represent values that you must supply.
example user input	In examples, information that the user enters appears in bold typeface.
output	Information that the system displays appears in this typeface.

- [] Square brackets enclose optional items in formats and command descriptions.
- { } Braces enclose a list from which you must choose an item in formats and command descriptions.
- | A vertical bar separates items in a list of choices.

Related Manuals

The file `/install/doc/apollo/os.v.latest software release number _manuals` lists current titles and revisions for all available manuals. For example, at SR10.0 refer to `/install/doc/apollo/os.v.10.0 _manuals` to check that you are using the correct version of manuals. You may also want to use this file to check that you have ordered all of the manuals that you need. (If you are using the Aegis environment, you can access the same information through the Help system by typing `help manuals`.)

Refer to the *Domain Documentation Quick Reference* (002685) and the *Domain Documentation Master Index* (011242) for a complete list of related documents. Refer to the following documents for more information on Domain@/OS, Aegis, BSD, and SysV:

<i>Getting Started with Domain/OS</i>	(2348)
<i>Domain/OS Display Manager Command Reference</i>	(11418)
<i>Domain/OS Programming Environment Reference</i>	(11010)
<i>Domain/OS Call Reference, Volume 1</i>	(7196)
<i>Domain/OS Call Reference, Volume 2</i>	(12888)
<i>Managing Domain Routing and Domain/OS in an Internet</i>	(5694)
<i>Domain Distributed Debugging Environment</i>	(11024)
<i>Using the Open Systems Toolkit to Extend the Streams Facility</i>	(863)
<i>DPSS/Mail User's Guide</i>	(3660)
<i>Writing Device Drivers with GPIO Calls</i>	(959)
<i>Using Your Aegis Environment</i>	(11021)
<i>Managing Aegis System Software</i>	(10852)

<i>Using Your BSD Environment</i>	(11020)
<i>Using Your SysV Environment</i>	(11022)
<i>BSD Command Reference</i>	(5800)
<i>SysV Command Reference</i>	(5798)

Problems, Questions, and Suggestions

We appreciate comments from the people who use our system. In order to make it easy for you to communicate with us, we provide the Apollo® Product Reporting (APR) system for software-related comments, and the Reader's Response form for documentation comments. By using these formal channels you make it easy for us to respond to your comments.

You can get more information about how to submit an APR by consulting the *Domain System Command Reference*. Refer to the `mkapr` shell command description. You can view the same description online by typing:

```
$ help mkapr
```

For your documentation comments, we've included a Reader's Response form at the back of each manual.

Contents

Chapter 1	Shell Basics	
1.1	Command Format	1-1
1.1.1	Arguments	1-1
1.1.2	Separators	1-2
1.1.3	Node Specifications	1-2
1.2	Using Special Characters	1-4
1.3	The Command Line Parser	1-6
1.3.1	Standard Command Options	1-7
1.3.2	Command Line Parser Options	1-7

Chapter 2	Aegis Commands	
abtsevset or display the abort-severity level	
acllist or copy an ACL	
aqdevacquire control of a PBU device	
arcfmaintain an archive file	
argsecho command line arguments	
bindcombine object modules into an executable file	
bltddisplay time operating system was built	
boffdeactivate the shell's -b flag	
bonactivate the shell's -b flag	
calendarset system calendar clock	
catfread file(s) and write to standard output	
chhdirchange a log-in home directory	
chnchange an object's name	
chpasschange a log-in password	
chpatreplace pattern in text file	
cmfidentify differences among files	
cmsrffind lines common to two files	
cmtcompare source tree to target tree	
cpbootcopy the system boot file sysboot	
cpfcopy a file	
cplcopy a link	

cpscr	copy the current display to a file
cpt	copy a directory tree
crd	create a directory
crddf	create, display, or modify a device descriptor file
crefs	cross-reference symbols in a file
crf	create a file
crl	create a link
crp	create a process on a remote node
crpad	create a transcript pad and window
crsubs	create a protected subsystem
crty	create a new type
crtyobj	create a type object module for binding
csr	set or display command search rules
ctnode	catalog a node in the network
ctob	catalog an object
cvt_font	convert fonts from pre-SR10 to SR10 format
cvt_rec_uasc	convert file types
cvtname	convert pathnames between upper and lowercase and preserve colons
cvtrgy	convert registry between SR9.x and SR10 formats
date	display the current date and time
dcalc	evaluate logical and arithmetic expressions
dde	Domain Distributed Debugging Environment
dldupl	strip repeated lines from a file
dlf	delete one or more files
dll	delete a link
dlt	delete a tree
dltv	delete a type
dlvar	deletes all of the specified variables
dmtvol	dismount a logical volume
drm_admin	Data Replication Manager Administrative Tool
dspst	display process status graphically
dtcb	dump contents of tcp control blocks
ed	invoke line editor
edacl	edit or list an ACL
edfont	edit a character font
edmtdesc	edit magtape descriptor file
edns	invoke editor for ns helper
edrgy	edit the network registry database
edsd	edit mail subscriber directory
edstr	edit a stream
em3270	emulate an IBM 3270 terminal
emt	emulate a dumb terminal
ensubs	enter a protected subsystem environment
enquire	inquire about system environment
eoff	deactivate the shell's -e flag
eon	activate the shell's -e flag
eqs	compare strings for equality
esa	display address of external symbol

exfld..... manipulate fields of data
 existf..... check for existence of an object
 existvar..... check that a variable is set
 exit..... exit from a loop
 export change a shell variable into an environment variable
 find_orphans..... locate and catalog uncataloged objects
 flen count lines, words, and characters in a file
 fmc format text into multiple columns
 fmt format a text file
 for..... execute a for statement
 fpat..... find a text pattern in an ASCII file
 fpatb..... find blocks of text containing patterns
 french_to_iso..... convert files to ISO format
 fserr..... find spelling errors
 fst print fault status information
 ftp..... ARPANET file transfer program
 german_to_iso..... convert files to ISO format
 glbd Global Location Broker Daemon
 help provide help on shell and DM commands
 hlpver provide help support for shell scripts
 hpc program counter histogram
 ifexecute a conditional statement
 import_passwd create registry entries based on group and password files
 inlib install a user-supplied library
 intm.....install a type manager
 inty install a new type
 invol initialize disk volumes
 ios_test..... test ios_\$ calls
 iso convert files to ISO format
 kbm set/display keyboard characteristics
 lamf..... laminate files
 las list objects mapped into the address space
 lb_admin Location Broker Administrative Tool
 lbr combine object modules into a library
 lbr2ar convert lbr libraries to SR10 archive libraries
 lcm..... load a color map
 lcnct display internet routing information
 lcnode list nodes connected to the network
 ldlist contents of a directory
 lkob lock an object
 llbd Local Location Broker Daemon
 llib list installed libraries
 llkob list locked objects
 lopstr list open streams
 lprotect..... control local protection
 lst..... list contents of a storage tree
 lty list installed types
 lusr list logged on users

lvar.....	list information about set variables
lvofls.....	list free space on logical volumes
macro.....	expand macro definitions
mbd.....	dump usage info on tcp buffer pool
mkapr.....	make an Apollo product report
mkcon.....	set console device
mkdev.....	shell script to make devices
mtvol.....	mount a logical volume
mvf.....	move a file
nd.....	set or display naming directory
netmain.....	analyze network maintenance stats
netmain_chklog.....	clean up bad log files
netmain_note.....	place message in network error log
netmain_srvr.....	collect network error stats
netstat.....	display network statistics
netsvc.....	set or display network services
next.....	return to the top of a loop
nor.dan_to_iso.....	convert files to ISO format
not.....	negate a Boolean value
obj2coff.....	convert OBJ format modules to COFF format modules
obty.....	set or display the type of an object
os.....	convert ASCII to FORTRAN carriage control
pagf.....	paginate a file
ppri.....	set or display process priority
prf.....	queue a file for printing by Domain/OS Aegis print spooler
prmgr.....	start the print manager
probenet.....	probe network and display error statistics
prsvr.....	start the print server
pst.....	list process internal state information
rbak.....	restore or index a magnetic media backup file
rdym.....	set system ready message
read.....	set variables equal to input values
readc.....	set variables equal to input characters
readln.....	set a variable equal to an input value
return.....	return from current shell level
revl.....	reverse each line in a file
rgy_admin.....	registry server administrative tool
rgy_create.....	registry creation utility
rgy_merge.....	merge registry database
rgyd.....	network registry server
rldev.....	release device acquired with aqdev
rtchk.....	test traffic between adjacent routers
rtstat.....	display internet router information
rtsvc.....	set or display internet routing service
rwmt.....	read/write foreign magtapes
salacl.....	salvage an access control list
sald.....	salvage a directory
salvol.....	verify and correct allocation of disk blocks

scrattr screen attributes
 scrtio set/show screen timeout
 select execute a select statement
 send_alarm send messages to alarm servers
 server run a server process
 set set current shell conditions
 setvar set the value of a variable
 sh invoke a shell, command line interpreter
 show_lc shell script to indicate obsoleted system calls in a binary file
 shutspm shut down SPM on a node
 sigp signal a process
 siorf receive a file from a remote host
 siof transmit a file to a remote host
 source execute a shell script at the current shell level
 srf sort and/or merge text files
 stcode translate status code value to text message
 subs set or display subsystem attributes
 swedish_to_iso convert files to ISO format
 swiss_to_iso convert files to ISO format
 syncids fix or verify file owners in a file system
 tb print process traceback
 tcpstat show network status
 tcrl set or display SIO line characteristics
 tee copy input to output and to named files
 telnet user interface to the TELNET protocol
 tlc replace characters
 tpm set/display touchpad and mouse characteristics
 tr_font transliterate characters within a font
 ts display the module name and time stamp
 tz set or display system time zone
 uctnode uncatlog a node
 uctob uncatlog the specified pathname, without deleting the associated object
 uk_to_iso convert files to ISO format
 ulkob unlock an object
 umask set UNIX file-creation-mode mask
 uuid_gen UUID generating program
 vctl set/display VT100 terminal characteristics
 voff deactivate the shell's -v flag
 von activate the shell's -v flag
 vsize set/display VT100 window settings
 vt100 VT100 terminal emulator
 wbak create a magnetic media backup file
 wd set or display the current working directory
 while execute a while loop
 xdmc execute a DM command from the shell
 xoff deactivate the shell's -x flag
 xon activate the shell's -x flag
 xsubs run shell-script subsystem manager

PERMUTED INDEX

<p>em3270 emulate an IBM abtsev set or display the abort-severity level salacl salvage an acl list or copy an edacl edit or list an device aqdev rlddev release device bon eon von xon esa display list objects mapped into the rtchk test traffic between /Data Replication Manager lb_admin Location Broker rgy_admin registry server file for printing by Domain/OS send_alarm send messages to salvol verify and correct stats netmain srf sort mkapr make an PBU device release device acquired with file arcf maintain an convert lbr libraries to SR10 arguments args echo command line dcalc evaluate logical and ftp find a text pattern in an control os convert pathname, without deleting the scrattr screen subs set or display subsystem boff deactivate the shell's bon activate the shell's or index a magnetic media wbak create a magnetic media netmain_chklog clean up /create registry entries obsolete system calls in a into an executable file a type object module for system was built dump contents of tcp control patterns fpatb find and correct allocation of disk -b flag flag</p>	<p>3270 terminal em3270 abort-severity level abtsev abtsev set or display the abtsev access control list salacl acl list or copy an ACL acl ACL acl ACL edacl acquire control of a PBU aqdev acquired with aqdev rlddev activate the shell's -b flag bon activate the shell's -e flag eon activate the shell's -v flag von activate the shell's -x flag xon address of external symbol esa address space las las adjacent routers rtchk Administrative Tool drm_admin Administrative Tool lb_admin administrative tool rgy_admin Aegis print spooler /queue a prf alarm servers send_alarm allocation of disk blocks salvol analyze network maintenance netmain and/or merge text files srf Apollo product report mkapr aqdev acquire control of a aqdev aqdev rlddev rlddev arcf maintain an archive arcf archive file arcf archive libraries lbr2ar lbr2ar args echo command line args arguments args arithmetic expressions dcalc ARPANET file transfer program ftp ASCII file fpat fpat ASCII to FORTRAN carriage os associated object /specified uctob attributes scrattr attributes subs -b flag boff -b flag bon backup file rbak restore rbak backup file wbak bad log files netmain_chklog based on group and password/ import_passwd binary file /to indicate show_lc bind combine object modules bind binding crtyobj create crtyobj bldt display time operating bldt blocks dtcb dtcb blocks of text containing fpatb blocks salvol verify salvol boff deactivate the shell's boff bon activate the shell's -b bon</p>
---	---

not negate a	Boolean value	not
cpboot copy the system	boot file sysboot	cpboot
lb_admin Location	Broker Administrative Tool	lb_admin
glbd Global Location	Broker Daemon	glbd
llbd Local Location	Broker Daemon	llbd
mbd dump usage info on tcp	buffer pool	mbd
time operating system was	built bldt display	bldt
clock	calendar set system calendar	calendar
calendar set system	calendar clock	calendar
/to indicate obsoleted system	calls in a binary file	show_lc
ios_test test ios_\$	calls	ios_test
os convert ASCII to FORTRAN	carriage control	os
ctnode	catalog a node in the network	ctnode
ctob	catalog an object	ctob
find_orphans locate and	catalog uncataloged objects	find_orphans
to standard output	catf read file(s) and write	catf
edfont edit a	character font	edfont
kbm set/display keyboard	characteristics	kbm
ctl set or display SIO line	characteristics	ctl
set/display touchpad and mouse	characteristics tpm	tpm
set/display VT100 terminal	characteristics vctl	vctl
flen count lines, words, and	characters in a file	flen
set variables equal to input	characters readc	readc
tlc replace	characters	tlc
tr_font transliterate	characters within a font	tr_font
object existf	check for existence of an	existf
existvar	check that a variable is set	existvar
directory	chmdir change a log-in home	chmdir
password	chn change an object's name	chn
file	chpass change a log-in	chpass
netmain_chklog	chpat replace pattern in text	chpat
calendar set system calendar	clean up bad log files	netmain_chklog
among files	clock	calendar
two files	cmf identify differences	cmf
target tree	cmsrf find lines common to	cmsrf
stcode translate status	cmt compare source tree to	cmt
/convert OBJ format modules to	code value to text message	stcode
netmain_srvr	COFF format modules	obj2coff
and lowercase and preserve	collect network error stats	netmain_srvr
lcm load a	colons /between upper	cvtname
fmc format text into multiple	color map	lcm
library lbr	columns	fmc
executable file bind	combine object modules into a	lbr
xdmc execute a DM	combine object modules into an	bind
args echo	command from the shell	xdmc
sh invoke a shell,	command line arguments	args
csr set or display	command line interpreter	sh
provide help on shell and DM	command search rules	csr
cmdsrf find lines	commands help	help
tree cmt	common to two files	cmdsrf
eqs	compare source tree to target	cmt
if execute a	compare strings for equality	eqs
set set current shell	conditional statement	if
lcnode list nodes	conditions	set
mkcon set	connected to the network	lcnode
	console device	mkcon

fpafb	find blocks of text containing patterns	fpafb
ld	list contents of a directory	ld
lst	list contents of a storage tree	lst
blocks dtcb	dump contents of tcp control	dtcb
dtcb	dump contents of tcp control blocks	dtcb
salact	salvage an access control list	salact
lprotect	control local protection	lprotect
aqdev	control of a PBU device	aqdev
ASCII to FORTRAN	carriage control os convert	os
carriage control	os convert ASCII to FORTRAN	os
cvt_rec_uasc	convert file types	cvt_rec_uasc
french_to_iso	convert files to ISO format	french_to_iso
german_to_iso	convert files to ISO format	german_to_iso
iso	convert files to ISO format	iso
nor.dan_to_iso	convert files to ISO format	nor.dan_to_iso
swedish_to_iso	convert files to ISO format	swedish_to_iso
swiss_to_iso	convert files to ISO format	swiss_to_iso
uk_to_iso	convert files to ISO format	uk_to_iso
SR10 format	cvt_font convert fonts from pre-SR10 to	cvt_font
archive libraries	lbr2ar convert lbr libraries to SR10	lbr2ar
COFF format modules	obj2coff convert OBJ format modules to	obj2coff
upper and lowercase/	cvtname convert pathnames between	cvtname
and SR10 formats	cvtrgy convert registry between SR9.x	cvtrgy
cpt	copy a directory tree	cpt
cpf	copy a file	cpf
cpl	copy a link	cpl
acl list or	copy an ACL	acl
named files	tee copy input to output and to	tee
file cpscr	copy the current display to a	cpscr
sysboot	cpboot copy the system boot file	cpboot
blocks salvol	verify and correct allocation of disk	salvol
characters in a file	flen count lines, words, and	flen
hpc	program counter histogram	hpc
file sysboot	cpboot copy the system boot	cpboot
	cpf copy a file	cpf
	cpl copy a link	cpl
display to a file	cpscr copy the current	cpscr
	cpt copy a directory tree	cpt
	crd create a directory	crd
modify a device descriptor/	crddf create, display, or	crddf
crd	create a directory	crd
crf	create a file	crf
crl	create a link	crl
file wbak	create a magnetic media backup	wbak
crty	create a new type	crty
node crp	create a process on a remote	crp
crsubs	create a protected subsystem	crsubs
window crpad	create a transcript pad and	crpad
for binding	crtyobj create a type object module	crtyobj
device descriptor file	crddf create, display, or modify a	crddf
on group and/	import_passwd create registry entries based	import_passwd
rgy_create	registry creation utility	rgy_create
in a file	crefs cross-reference symbols	crefs
crf	create a file	crf
crl	create a link	crl
file crefs	cross-reference symbols in a	crefs

remote node	crp create a process on a	crp
and window	crpad create a transcript pad	crpad
subsystem	crsubs create a protected	crsubs
	crty create a new type	crty
module for binding	crtyobj create a type object	crtyobj
search rules	csr set or display command	csr
network	ctnode catalog a node in the	ctnode
	ctob catalog an object	ctob
date display the	current date and time	date
cpscr copy the	current display to a file	cpscr
set set	current shell conditions	set
return return from	current shell level	return
execute a shell script at the	current shell level source	source
wd set or display the	current working directory	wd
pre-SR10 to SR10 format	cvt_font convert fonts from	cvt_font
between upper and lowercase/	cvtname convert pathnames	cvtname
types	cvt_rec_uasc convert file	cvt_rec_uasc
between SR9.x and SR10/	cvtrgy convert registry	cvtrgy
glbd Global Location Broker	Daemon	glbd
llbd Local Location Broker	Daemon	llbd
exfld manipulate fields of	data	exfld
Administrative/ drm_admin	Data Replication Manager	drm_admin
edit the network registry	database edrgy	edrgy
rgy_merge merge registry	database	rgy_merge
and time	date display the current date	date
date display the current	date and time	date
arithmetic expressions	dcalc evaluate logical and	dcalc
Debugging Environment	dde Domain Distributed	dde
flag boff	deactivate the shell's -b	boff
flag eoff	deactivate the shell's -e	eoff
flag voff	deactivate the shell's -v	voff
flag xoff	deactivate the shell's -x	xoff
dde Domain Distributed	Debugging Environment	dde
macro expand macro	definitions	macro
	delete a link	dll
	delete a tree	dlt
	delete a type	dltty
	delete one or more files	dlf
variables dlvvar	deletes all of the specified	dlvar
specified pathname, without	deleting the associated/the	uctob
display, or modify a device	descriptor file /create,	crddf
edmtdesc edit magtape	descriptor file	edmtdesc
rlddev release	device acquired with aqdev	rlddev
acquire control of a PBU	device aqdev	aqdev
/create, display, or modify a	device descriptor file	crddf
mkcon set console	device	mkcon
mkdev shell script to make	devices	mkdev
cmf identify	differences among files	cmf
chhdir change a log-in home	directory	chhdir
crd create a	directory	crd
edsd edit mail subscriber	directory	edsd
ld list contents of a	directory	ld
nd set or display naming	directory	nd
sald salvage a	directory	sald
cpt copy a	directory tree	cpt
or display the current working	directory wd set	wd

and correct allocation of	disk blocks	salvol	verify	salvol
invol	initialize	disk	volumes	invol
dmtvol		dismount	a logical volume	dmtvol
symbol	esa	display	address of external	esa
csr	set or	display	command search rules	csr
probenet	probe network and	display	error statistics	probenet
information	rtstat	display	internet router	rtstat
information	lcnnet	display	internet routing	lcnnet
service	rtsvc set or	display	internet routing	rtsvc
nd	set or	display	naming directory	nd
netsvc	set or	display	network services	netsvc
netstat		display	network statistics	netstat
descriptor/	crddf create,	display, or modify	a device	crddf
ppri	set or	display	process priority	ppri
graphically	dspst	display	process status	dspst
characteristics	tctl set or	display	SIO line	tctl
subs	set or	display	subsystem attributes	subs
tz	set or	display	system time zone	tz
level	abtsev set or	display	the abort-severity	abtsev
time	date	display	the current date and	date
directory	wd set or	display	the current working	wd
time stamp	ts	display	the module name and	ts
obty	set or	display	the type of an object	obty
was built	bltd	display	time operating system	bltd
cpscrc	copy the current	display	to a file	cpscrc
Environment	dde Domain	Distributed	Debugging	dde
from a file		dlDupl	strip repeated lines	dlDupl
		dlf	delete one or more files	dlf
		dll	delete a link	dll
		dlt	delete a tree	dlt
		dlt	delete a type	dlt
specified variables		dlvar	deletes all of the	dlvar
xdmc	execute a	DM	command from the shell	xdmc
provide help on shell and	volume	DM	commands help	help
Environment	dde	dmtvol	dismount a logical	dmtvol
/queue a file for printing by	Manager Administrative Tool	Domain	Distributed Debugging	dde
graphically		Domain/OS	Aegis print spooler	prf
control blocks		drm_admin	Data Replication	drm_admin
emt	emulate a	dspst	display process status	dspst
blocks	dtcb	dtcb	dump contents of tcp	dtcb
pool	mbd	dumb	terminal	emt
coeff	deactivate the shell's	dump	contents of tcp control	dtcb
eon	activate the shell's	dump	usage info on tcp buffer	mbd
args		-e	flag	coeff
		-e	flag	eon
		echo	command line arguments	args
		ed	invoke line editor	ed
		edacl	edit or list an ACL	edacl
		edfont	edit a character font	edfont
		edstr	edit a stream	edstr
		edmtdesc	edit magtape descriptor file	edmtdesc
directory	edsd	edit	mail subscriber	edsd
		edit	or list an ACL	edacl
database	edrgy	edit	the network registry	edrgy
ed	invoke line	editor		ed

edns invoke	editor for ns_helper	edns
descriptor file	edmtdesc edit magtape	edmtdesc
ns_helper	edns invoke editor for	edns
registry database	edrgy edit the network	edrgy
directory	edsd edit mail subscriber	edsd
	edstr edit a stream	edstr
terminal	em3270 emulate an IBM 3270	em3270
	emt emulate a dumb terminal	emt
	emt emulate a dumb terminal	emt
	em3270 emulate an IBM 3270 terminal	em3270
vt100 VT100 terminal	emulator	vt100
subsystem	ensubs enter a protected	ensubs
ensubs	enter a protected subsystem	ensubs
import_passwd create registry	entries based on group and/	import_passwd
system environment	environment inquire about	environment
Domain Distributed Debugging	Environment dde	dde
inquire about system	environment environment	environment
a shell variable into an	environment variable /change	export
-e flag	eff deactivate the shell's	eff
flag	eon activate the shell's -e	eon
equality	eqs compare strings for	eqs
readln set a variable	equal to an input value	readln
readc set variables	equal to input characters	readc
read set variables	equal to input values	read
eqs compare strings for	equality	eqs
place message in network	error log netmain_note	netmain_note
probe network and display	error statistics probenet	probenet
netmain_svr collect network	error stats	netmain_svr
fserr find spelling	errors	fserr
external symbol	esa display address of	esa
arithmetic expressions dcalc	evaluate logical and	dcalc
combine object modules into an	executable file bind	bind
statement if	execute a conditional	if
shell xdmc	execute a DM command from the	xdmc
for	execute a for statement	for
select	execute a select statement	select
current shell level source	execute a shell script at the	source
while	execute a while loop	while
data	exfld manipulate fields of	exfld
existf check for	existence of an object	existf
an object	existf check for existence of	existf
variable is set	existvar check that a	existvar
	exit exit from a loop	exit
	exit from a loop	exit
macro	expand macro definitions	macro
variable into an environment/	export change a shell	export
logical and arithmetic	expressions dcalc evaluate	dcalc
esa display address of	external symbol	esa
fst print	fault status information	fst
arcf maintain an archive	file	arcf
modules into an executable	file bind combine object	bind
chpat replace pattern in text	file	chpat
cpf copy a	file	cpf
copy the current display to a	file cpscr	cpscr
or modify a device descriptor	file crddf create, display,	crddf
cross-reference symbols in a	file crefs	crefs

	crf	create a file	crf
strip	repeated lines from a file	dldupl	dldupl
edit	magtape descriptor	edmtdesc	edmtdesc
words, and characters in a file	flen	count lines,	flen
fmt	format a text file	fmt	fmt
Aegis print/ prf	queue a file for printing by Domain/OS	prf	prf
a text pattern in an ASCII file	fpaf	find	fpaf
siorf	receive a file from a remote host	siorf	siorf
mvf	move a file	mvf	mvf
syncids	fix or verify file owners in a file system	syncids	syncids
pagf	paginate a file	pagf	pagf
index a magnetic media backup	rbak	restore or	rbak
revl	reverse each line in a file	revl	revl
system calls in a binary file	/to	indicate obsoleted	show_lc
cpboot	copy the system boot file	sysboot	cpboot
fix or verify file owners in a file system	syncids	syncids	syncids
siotf	transmit a file to a remote host	siotf	siotf
ftp	ARPANET file transfer program	ftp	ftp
cvt_rec_uasc	convert file types	cvt_rec_uasc	cvt_rec_uasc
create a magnetic media backup	wbak	wbak	wbak
umask	set UNIX file-creation-mode mask	umask	umask
output	catf	read file(s) and write to standard	catf
identify differences among files	cmf	cmf	cmf
find lines common to two files	cmsrf	cmsrf	cmsrf
dlf	delete one or more files	dlf	dlf
based on group and password files	/registry	entries	import_passwd
lamf	laminates files	lamf	lamf
clean up bad log files	netmain_chklog	netmain_chklog	netmain_chklog
srf	sort and/or merge text files	srf	srf
input to output and to named files	tee	copy	tee
french_to_iso	convert files to ISO format	french_to_iso	french_to_iso
german_to_iso	convert files to ISO format	german_to_iso	german_to_iso
iso	convert files to ISO format	iso	iso
nor.dan_to_iso	convert files to ISO format	nor.dan_to_iso	nor.dan_to_iso
swedish_to_iso	convert files to ISO format	swedish_to_iso	swedish_to_iso
swiss_to_iso	convert files to ISO format	swiss_to_iso	swiss_to_iso
uk_to_iso	convert files to ISO format	uk_to_iso	uk_to_iso
ASCII file	fpaf	find a text pattern in an	fpaf
patterns	fpafb	find blocks of text containing	fpafb
files	cmsrf	find lines common to two	cmsrf
fserr	find spelling errors	fserr	fserr
catalog uncataloged objects	find_orphans	locate and	find_orphans
file system	syncids	fix or verify file owners in a	syncids
deactivate the shell's -b	flag	boff	boff
bon	activate the shell's -b	flag	bon
deactivate the shell's -e	flag	eoff	eoff
eon	activate the shell's -e	flag	eon
deactivate the shell's -v	flag	voff	voff
von	activate the shell's -v	flag	von
deactivate the shell's -x	flag	xoff	xoff
xon	activate the shell's -x	flag	xon
characters in a file	flen	count lines, words, and	flen
columns	fmc	format text into multiple	fmc
	fmt	format a text file	fmt
edfont	edit a character font	edfont	edfont
characters within a font	tr_font	transliterate	tr_font

format	cvt_font convert	fonts from pre-SR10 to SR10	cvt_font
	rwmt read/write	foreign magtapes	rwmt
	fint	format a text file	fint
fonts from pre-SR10 to SR10	format cvt_font convert	cvt_font	
	convert files to ISO	format french_to_iso	french_to_iso
	convert files to ISO	format german_to_iso	german_to_iso
iso	convert files to ISO	format	iso
OBJ format modules to COFF	format modules /convert	obj2coff	
	obj2coff convert OBJ	format modules to COFF format/	obj2coff
	convert files to ISO	format nor.dan_to_iso	nor.dan_to_iso
	convert files to ISO	format swedish_to_iso	swedish_to_iso
	convert files to ISO	format swiss_to_iso	swiss_to_iso
	columns fmc	format text into multiple	fmc
	convert files to ISO	format uk_to_iso	uk_to_iso
between SR9.x and SR10	formats /convert registry	cvtrgy	
os	convert ASCII to	FORTAN carriage control	os
	an ASCII file	fpaf find a text pattern in	fpaf
	containing patterns	fpafb find blocks of text	fpafb
	lvofls list	free space on logical volumes	lvofls
	to ISO format	french_to_iso convert files	french_to_iso
dldupl	strip repeated lines	from a file	dldupl
	exit exit	from a loop	exit
	siorf receive a file	from a remote host	siorf
	return return	from current shell level	return
	cvt_font convert fonts	from pre-SR10 to SR10 format	cvt_font
xdmc	execute a DM command	from the shell	xdmc
	information	fserr find spelling errors	fserr
	program	fst print fault status	fst
	uuid_gen UUID	ftp ARPANET file transfer	ftp
	to ISO format	generating program	uuid_gen
	Daemon	german_to_iso convert files	german_to_iso
	gldb	gldb Global Location Broker	gldb
dspst	display process status	Global Location Broker Daemon	gldb
	/registry entries based on	graphically	dspst
	and DM commands	group and password files	import_passwd
	help provide	help provide help on shell	help
	scripts hlpver provide	help on shell and DM commands	help
	hpc program counter	help support for shell	hlpver
	for shell scripts	histogram	hpc
	chhdir change a log-in	hlpver provide help support	hlpver
	receive a file from a remote	home directory	chhdir
	transmit a file to a remote	host siorf	siorf
	histogram	host siof	siof
	em3270 emulate an	hpc program counter	hpc
	files cmf	IBM 3270 terminal	em3270
entries based on group and/	identify differences among	import_passwd create registry	import_passwd
	file rbak restore or	index a magnetic media backup	rbak
	show_lc shell script to	indicate obsoleted system/	show_lc
	mbd dump usage	info on tcp buffer pool	mbd
	invol	initialize disk volumes	invol
	library	inlib install a user-supplied	inlib
readc	set variables equal to	input characters	readc
	files tee copy	input to output and to named	tee
	set a variable equal to an	input value readln	readln
	read	input values	read

in a file	flen	count	lines, words, and characters	flen
	cpl	copy a	link	cpl
	crf	create a	link	crf
	dll	delete a	link	dll
	edacl	edit or	list an ACL	edacl
	ld		list contents of a directory	ld
	tree	lst	list contents of a storage	lst
volumes	lvofls		list free space on logical	lvofls
variables	lvar		list information about set	lvar
	llib		list installed libraries	llib
	lty		list installed types	lty
	llkob		list locked objects	llkob
	lusr		list logged on users	lusr
network	lcnode		list nodes connected to the	lcnode
address space	las		list objects mapped into the	las
	lopstr		list open streams	lopstr
	acl		list or copy an ACL	acl
information	pst		list process internal state	pst
salvage an access control	salacl		list salacl	salacl
	lkob	lock an object		lkob
Daemon	llbd	Local Location Broker		llbd
libraries	llib	list installed		llib
	llkob	list locked objects		llkob
	lcm	load a color map		lcm
objects	find_orphans	locate and catalog uncataloged		find_orphans
Tool	lb_admin	Location Broker Administrative		lb_admin
	glbd	Global Location Broker Daemon		glbd
	llbd	Local Location Broker Daemon		llbd
	lkob	lock an object		lkob
	llkob	list locked objects		llkob
netmain_chklog	clean up bad	log files		netmain_chklog
place message in network error	log netmain_note			netmain_note
	lusr	list logged on users		lusr
expressions	dcalc	evaluate logical and arithmetic		dcalc
	dmtvol	dismount a logical volume		dmtvol
	mtvol	mount a logical volume		mtvol
lvofls	list free space on	logical volumes		lvofls
	chhdir	change a log-in home directory		chhdir
	chpass	change a log-in password		chpass
	exit	exit from a loop		exit
next	return to the top of a	loop		next
while	execute a while	loop		while
	lopstr	list open streams		lopstr
/pathnames	between upper and	lowercase and preserve colons		cvtname
protection	lprotect	control local		lprotect
storage tree	lst	list contents of a		lst
	lty	list installed types		lty
	lusr	list logged on users		lusr
set variables	lvar	list information about		lvar
logical volumes	lvofls	list free space on		lvofls
definitions	macro	expand macro		macro
macro	expand	macro definitions		macro
rbak	restore or index a	magnetic media backup file		rbak
	wbak	create a magnetic media backup file		wbak
	edmtdesc	edit magtape descriptor file		edmtdesc
rwmt	read/write foreign	magtapes		rwmt

	edsd edit	mail subscriber directory	edsd
	arcf	maintain an archive file	arcf
netmain	analyze network	netmain
	mkapr	make an Apollo product report	mkapr
	mkdev	shell script to	mkdev
drm_admin	Data Replication	Manager Administrative Tool	drm_admin
	intm	install a type	intm
	pmgr	start the print	pmgr
run	shell-script subsystem	manager xsubs	xsubs
	exfld	manipulate fields of data	exfld
	lcm	load a color	lcm
	las	list objects	las
set	UNIX file-creation-mode	mask umask	umask
	buffer pool	mbd dump usage info on tcp	mbd
restore	or index a magnetic	media backup file rbak	rbak
	wbak	create a magnetic	wbak
	rgy_merge	media backup file	rgy_merge
	srf	merge registry database	rgy_merge
	netmain_note	merge text files	srf
	rdym	message in network error log	netmain_note
	status	message	rdym
	send_alarm	message stcode translate	stcode
	report	messages to alarm servers	send_alarm
	devices	mkapr make an Apollo product	mkapr
	crddf	mkcon set console device	mkcon
crtyobj	create, display, or	mkdev shell script to make	mkdev
	ts	modify a device descriptor/	crddf
	lbr	module for binding	crtyobj
file	bind combine object	module name and time stamp	ts
format	modules to COFF format	modules into a library	lbr
	obj2coff	modules into an executable	bind
	mtvol	modules /convert OBJ	obj2coff
tpm	set/display touchpad and	modules to COFF format/	obj2coff
	mvf	mount a logical volume	mtvol
	fmc	mouse characteristics	tpm
	copy	move a file	mvf
	nd	mtvol mount a logical volume	mtvol
	directory	multiple columns	fmc
	not	mvf move a file	mvf
	maintenance	named files tee	tee
	log	naming directory	nd
	network	nd set or display naming	nd
	error	negate a Boolean value	not
	stats	netmain analyze network	netmain
	services	netmain_chklog clean up bad	netmain_chklog
	statistics	netmain_note place message in	netmain_note
	probenet	netmain_srvr collect network	netmain_srvr
ctnode	catalog a node in the	netstat display network	netstat
	netmain_note	netsvc set or display network	netsvc
	netmain_srvr	network and display error	probenet
	collect	network	ctnode
	list	network error log	netmain_note
	nodes	network error stats	netmain_srvr
	connected	network lcnode	lcnode
	to the	network maintenance stats	netmain
	edrgy	network registry database	edrgy
	edit	network registry server	rgyd
	the		
	rgyd		

netnvc set or display	network services	netnvc
netstat display	network statistics	netstat
nS show	network status	tcpstat
create a process on a remote	node crp	crp
ctnode catalog a	node in the network	ctnode
shutspm shut down SPM on a	node	shutspm
uctnode uncatalog a	node	uctnode
network lcnode list	nodes connected to the	lcnode
to ISO format	nor.dan_to_iso convert files	nor.dan_to_iso
	nS show network status	tcpstat
ednS invoke editor for	ns_helper	ednS
format/ obj2coff convert	OBJ format modules to COFF	obj2coff
modules to COFF format/	obj2coff convert OBJ format	obj2coff
ctob catalog an	object	ctob
check for existence of an	object existf	existf
lkob lock an	object	lkob
crtyobj create a type	object module for binding	crtyobj
lbr combine	object modules into a library	lbr
executable/ bind combine	object modules into an	bind
set or display the type of an	object obty	obty
deleting the associated	object /pathname, without	uctob
ulkob unlock an	object	ulkob
locate and catalog uncataloged	objects find_orphans	find_orphans
lkob list locked	objects	lkob
address space las list	objects mapped into the	las
chn change an	object's name	chn
/shell script to indicate	obsoleted system calls in a/	show_lc
of an object	obty set or display the type	obty
lopstr list	open streams	lopstr
bldt display time	operating system was built	bldt
carriage control	os convert ASCII to FORTRAN	os
tee copy input to	output and to named files	tee
file(s) and write to standard	output catf read	catf
syncids fix or verify file	owners in a file system	syncids
crpad create a transcript	pad and window	crpad
	pagf paginate a file	pagf
pagf	paginate a file	pagf
password	password	chpass
chpass change a log-in	password files /registry	import_passwd
entries based on group and	pathname, without deleting the/	uctob
uctob uncatalog the specified	pathnames between upper and	cvtname
lowercase/ cvtname convert	pattern in an ASCII file	fpat
fpat find a text	pattern in text file	chpat
chpat replace	patterns fpatb	fpatb
find blocks of text containing	PBU device	aqdev
aqdev acquire control of a	pool mbd	mbd
dump usage info on tcp buffer	ppri set or display process	ppri
priority	preserve colons /between	cvtname
upper and lowercase and	pre-SR10 to SR10 format	cvt_font
cvt_font convert fonts from	prf queue a file for printing	prf
by Domain/OS Aegis print/	print fault status	fst
information fst	print manager	prmgr
prmgr start the	print process traceback	tb
tb	print server	prsvr
prsvr start the	print spooler /a file for	prf
printing by Domain/OS Aegis	printing by Domain/OS Aegis	prf
print/ prf queue a file for		

ppri set or display process manager	priority	ppri
error statistics	prmgr start the print	prmgr
display error statistics	probenet probe network and	probenet
information pst list	probenet probe network and	probenet
crp create a	process internal state	pst
ppri set or display	process on a remote node	crp
server run a server	process priority	ppri
sigp signal a	process	server
dspst display	process	sigp
tb print	process status graphically	dspst
mkapr make an Apollo	process traceback	tb
crsubs create a	product report	mkapr
ensubs enter a	protected subsystem	crsubs
lprotect control local	protected subsystem	ensubs
user interface to the TELNET	protection	lprotect
commands help	protocol telnet	telnet
scripts hlpver	provide help on shell and DM	help
	provide help support for shell	hlpver
	prsvr start the print server	prsvr
state information	pst list process internal	pst
Domain/OS Aegis print/ prf	queue a file for printing by	prf
magnetic media backup file	rbak restore or index a	rbak
message	rdym set system ready	rdym
input values	read set variables equal to	read
standard output catf	read file(s) and write to	catf
input characters	readc set variables equal to	readc
to an input value	readln set a variable equal	readln
	read/write foreign magtapes	rwmt
rwmt	ready message	rdym
rdym set system	receive a file from a remote	siorf
host siorf	registry between SR9.x and	cvtrgy
SR10 formats cvtrgy convert	registry creation utility	rgy_create
rgy_create	registry database	edrgy
edrgy edit the network	registry database	rgy_merge
rgy_merge merge	registry entries based on	import_passwd
group/ import_passwd create	registry server administrative	rgy_admin
tool rgy_admin	registry server	rgyd
rgyd network	release device acquired with	rldev
aqdev rldev	remote host	siorf
siorf receive a file from a	remote host	siof
siof transmit a file to a	remote node	crp
crp create a process on a	repeated lines from a file	dldupl
dldupl strip	replace characters	tlc
tlc	replace pattern in text file	chpat
chpat	Replication Manager/	drm_admin
drm_admin Data	report	mkapr
mkapr make an Apollo product	restore or index a magnetic	rbak
media backup file rbak	return return from current	return
shell level	return from current shell	return
level return	return to the top of a loop	next
next	reverse each line in a file	revl
revl	revl reverse each line in a	revl
file	rgy_admin registry server	rgy_admin
administrative tool	rgy_create registry creation	rgy_create
utility	rgyd network registry server	rgyd
database	rgy_merge merge registry	rgy_merge

with aqdev	ridev release device acquired	ridev
rtstat display internet	router information	rtstat
test traffic between adjacent	routers rtchk	rtchk
lcnnet display internet	routing information	lcnnet
rtsvc set or display internet	routing service	rtsvc
adjacent routers	rtchk test traffic between	rtchk
router information	rtstat display internet	rtstat
routing service	rtsvc set or display internet	rtsvc
set or display command search	rules csr	csr
server	run a server process	server
manager xsubs	run shell-script subsystem	xsubs
magtapes	rwmt read/write foreign	rwmt
control list	salacl salvage an access	salacl
	sald salvage a directory	sald
	salvage a directory	sald
list salacl	salvage an access control	salacl
allocation of disk blocks	salvol verify and correct	salvol
	scrattr screen attributes	scrattr
scrattr	screen attributes	scrattr
scrto set/show	screen timeout	scrto
source execute a shell	script at the current shell/	source
system calls/ show_lc shell	script to indicate obsoleted	show_lc
mkdev shell	script to make devices	mkdev
provide help support for shell	scripts hlpver	hlpver
timeout	scrto set/show screen	scrto
csr set or display command	search rules	csr
statement	select execute a select	select
select execute a	select statement	select
servers send_alarm	send messages to alarm	send_alarm
alarm servers	send_alarm send messages to	send_alarm
	server run a server process	server
rgy_admin registry	server administrative tool	rgy_admin
server run a	server process	server
prsvr start the print	server	prsvr
rgyd network registry	server	rgyd
send messages to alarm	servers send_alarm	send_alarm
characteristics kbm	set/display keyboard	kbm
characteristics tpm	set/display touchpad and mouse	tpm
characteristics vctl	set/display VT100 terminal	vctl
settings vsize	set/display VT100 window	vsize
scrto	set/show screen timeout	scrto
set/display VT100 window	settings vsize	vsize
variable	setvar set the value of a	setvar
line interpreter	sh invoke a shell, command	sh
help provide help on	shell and DM commands	help
interpreter sh invoke a	shell, command line	sh
set set current	shell conditions	set
return return from current	shell level	return
a shell script at the current	shell level source execute	source
shell/ source execute a	shell script at the current	source
obsoleted system/ show_lc	shell script to indicate	show_lc
mkdev	shell script to make devices	mkdev
provide help support for	shell scripts hlpver	hlpver
environment/ export change a	shell variable into an	export
execute a DM command from the	shell xdmc	xdmc
boff deactivate the	shell's -b flag	boff

bon activate the	shell's -b flag	bon
eoff deactivate the	shell's -e flag	eoff
eon activate the	shell's -e flag	eon
voff deactivate the	shell's -v flag	voff
von activate the	shell's -v flag	von
xoff deactivate the	shell's -x flag	xoff
xon activate the	shell's -x flag	xon
manager xsubs run	shell-script subsystem	xsubs
indicate obsoleted system/	show_lc shell script to	show_lc
shutspm	shut down SPM on a node	shutspm
node	shutspm shut down SPM on a	shutspm
sigp	signal a process	sigp
	sigp signal a process	sigp
tcl set or display	SIO line characteristics	tcl
remote host	siorf receive a file from a	siorf
remote host	siotf transmit a file to a	siotf
srf	sort and/or merge text files	srf
at the current shell level	source execute a shell script	source
cmt compare	source tree to target tree	cmt
mapped into the address	space las list objects	las
lvofls list free	space on logical volumes	lvofls
deleting/ uctob uncatlog the	specified pathname, without	uctob
dlvar deletes all of the	specified variables	dlvar
fserr find	spelling errors	fserr
shutspm shut down	SPM on a node	shutspm
by Domain/OS Aegis print	spooler /a file for printing	prf
/convert lbr libraries to	SR10 archive libraries	lbr2ar
convert fonts from pre-SR10 to	SR10 format cvt_font	cvt_font
registry between SR9.x and	SR10 formats cvtrgy convert	cvtrgy
/convert registry between	SR9.x and SR10 formats	cvtrgy
files	srf sort and/or merge text	srf
the module name and time	stamp ts display	ts
read file(s) and write to	standard output catf	catf
prmgr	start the print manager	prmgr
prsvr	start the print server	prsvr
for execute a for	statement	for
if execute a conditional	statement	if
select execute a select	statement	select
netstat display network	statistics	netstat
network and display error	statistics probenet probe	probenet
analyze network maintenance	stats netmain	netmain
collect network error	stats netmain_svr	netmain_svr
message stcode translate	status code value to text	stcode
dspst display process	status graphically	dspst
fst print fault	status information	fst
nS show network	status	tcpstat
value to text message	stcode translate status code	stcode
lst list contents of a	storage tree	lst
edstr edit a	stream	edstr
lopstr list open	streams	lopstr
eqs compare	strings for equality	eqs
file dldupl	strip repeated lines from a	dldupl
attributes	subs set or display subsystem	subs
edsd edit mail	subscriber directory	edsd
subs set or display	subsystem attributes	subs
crsubs create a protected	subsystem	crsubs

ensubs enter a protected subsystem	ensubs
xsubs run shell-script subsystem manager	xsubs
hlpver provide help support for shell scripts	hlpver
to ISO format swedish_to_iso convert files	swedish_to_iso
ISO format swiss_to_iso convert files to	swiss_to_iso
display address of external symbol esa	esa
crefs cross-reference symbols in a file	crefs
owners in a file system syncids fix or verify file	syncids
copy the system boot file sysboot cpboot	cpboot
cmt compare source tree to target tree	cmt
tb print process traceback	tb
mbd dump usage info on tcp buffer pool	mbd
dtcb dump contents of tcp control blocks	dtcb
characteristics tctl set or display SIO line	tctl
to named files tee copy input to output and	tee
TELNET protocol telnet user interface to the	telnet
telnet user interface to the TELNET protocol	telnet
vctl set/display VT100 terminal characteristics	vctl
em3270 emulate an IBM 3270 terminal	em3270
emt emulate a dumb terminal	emt
vt100 VT100 terminal emulator	vt100
ios_test test ios_\$ calls	ios_test
routers rtchk test traffic between adjacent	rtchk
fpatb find blocks of text containing patterns	fpatb
chpat replace pattern in text file	chpat
fmt format a text file	fmt
srf sort and/or merge text files	srf
fmc format text into multiple columns	fmc
translate status code value to text message stcode	stcode
fpat find a text pattern in an ASCII file	fpat
display the current date and time date	date
built bldt display time operating system was	bldt
display the module name and time stamp ts	ts
tz set or display system time zone	tz
scrto set/show screen timeout	scrto
Manager Administrative tlc replace characters	tlc
Location Broker Administrative Tool /Data Replication	drm_admin
registry server administrative Tool lb_admin	lb_admin
next return to the tool rgy_admin	rgy_admin
tpm set/display top of a loop	next
mouse characteristics touchpad and mouse/	tpm
tb print process tpm set/display touchpad and	tpm
routers rtchk test traceback	tb
cpad create a traffic between adjacent	rtchk
ftp ARPANET file transcript pad and window	cpad
text message stcode transfer program	ftp
within a font tr_font translate status code value to	stcode
host siotf transliterate characters	tr_font
compare source tree to target transmit a file to a remote	siotf
cpt copy a directory tree cmt	cmt
dlt delete a tree	cpt
list contents of a storage tree lst	dlt
cmt compare source tree to target tree	lst
characters within a font tr_font transliterate	cmt
and time stamp ts display the module name	tr_font
	ts

crty create a new	type	crty
dlty delete a	type	dlty
inty install a new	type	inty
intm install a	type manager	intm
binding crtyobj create a	type object module for	crtyobj
obty set or display the	type of an object	obty
cvt_rec_uasc convert file	types	cvt_rec_uasc
lty list installed	types	lty
zone	tz set or display system time	tz
	uctnode uncatlog a node	uctnode
pathname, without deleting/	uctob uncatlog the specified	uctob
ISO format	uk_to_iso convert files to	uk_to_iso
	ulkob unlock an object	ulkob
file-creation-mode mask	umask set UNIX	umask
uctnode	uncatlog a node	uctnode
pathname, without/ uctob	uncatlog the specified	uctob
/locate and catalog	uncataloged objects	find_orphans
ulkob	unlock an object	ulkob
/convert pathnames between	upper and lowercase and/	cvtname
mbd dump	usage info on tcp buffer pool	mbd
protocol telnet	user interface to the TELNET	telnet
lusr list logged on	users	lusr
inlib install a	user-supplied library	inlib
rgy_create registry creation	utility	rgy_create
uuid_gen	UUID generating program	uuid_gen
program	uuid_gen UUID generating	uuid_gen
voff deactivate the shell's	-v flag	voff
von activate the shell's	-v flag	von
not negate a Boolean	value	not
setvar set the	value of a variable	setvar
a variable equal to an input	value readln set	readln
stcode translate status code	value to text message	stcode
set variables equal to input	values read	read
value readln set a	variable equal to an input	readln
variable into an environment	variable /change a shell	export
export change a shell	variable into an environment/	export
existvar check that a	variable is set	existvar
setvar set the value of a	variable	setvar
deletes all of the specified	variables dlvvar	dlvvar
characters readc set	variables equal to input	readc
values read set	variables equal to input	read
list information about set	variables lvar	lvar
terminal characteristics	vctl set/display VT100	vctl
of disk blocks salvol	verify and correct allocation	salvol
system syncids fix or	verify file owners in a file	syncids
-v flag	voff deactivate the shell's	voff
dmtvol dismount a logical	volume	dmtvol
mtvol mount a logical	volume	mtvol
invol initialize disk	volumes	invol
list free space on logical	volumes lvolfs	lvolfs
flag	von activate the shell's -v	von
window settings	vsize set/display VT100	vsize
emulator	vt100 VT100 terminal	vt100
vctl set/display	VT100 terminal/	vctl
vt100	VT100 terminal emulator	vt100
vsize set/display	VT100 window settings	vsize

backup file	wbak create a magnetic media	wbak
working directory	wd set or display the current	wd
create a transcript pad and	window crpad	crpad
vsize set/display VT100	window settings	vsize
file flen count lines,	words, and characters in a	flen
wd set or display the current	working directory	wd
catf read file(s) and	write to standard output	catf
xoff deactivate the shell's	-x flag	xoff
xon activate the shell's	-x flag	xon
from the shell	xdmc execute a DM command	xdmc
-x flag	xoff deactivate the shell's	xoff
flag	xon activate the shell's -x	xon
subsystem manager	xsubs run shell-script	xsubs
tz set or display system time	zone	tz

Chapter 1

Shell Basics

This chapter summarizes the basic concepts that apply to the shell commands described individually in the following chapter. See *Using Your Aegis Environment* for a detailed discussion of these concepts.

1.1 Command Format

In the most general sense, the operating system has no commands. There are simply files that the shell looks for and executes. When you type "date" in the shell input window, the shell looks for a file called `date` (following its command search rules) and executes it. This means that you can give any files that you create to the shell for execution. Of course, if you tell the shell to execute a file containing non-binary data -- like the text of a memo -- you get an error message. The point is that you can give any file, no matter where it comes from, to the shell for interpretation and execution.

The simplest command line consists of a command name followed by arguments to the command, separated by spaces:

```
command arg1 arg2 ... argn
```

1.1.1 Arguments

The command shell, which we supply, handles commands that accept multiple arguments (see Figure 1-1). Usually, those arguments come in two forms: a path-name designating a file on which to operate or some other sort of literal string for manipulation, and instructions for special command action. Those arguments that specify special action are almost always optional, and are immediately preceded by a hyphen (-). The hyphen is necessary because these arguments often require secondary arguments of their own. The commands use the hyphen to interpret correctly where all the different arguments apply. These special arguments are labeled **OPTIONS** in the command descriptions in Chapter 2.

only the node ID, Domain/OS gets the network number from either your local cache or the `ns_helper` database. However, if you provide a complete internet address, Aegis attempts to locate the node only on the network you specify. Thus, if you specify an incorrect network number, Domain/OS looks for the node only on the network that you specify and then reports an error; Domain/OS does not attempt to locate the node on another network.

If a node is not cataloged, Domain/OS cannot get a network number if you omit it. In this case, Domain/OS assumes that the node is on the local ring. Thus, for an uncataloged node on the local network, you must provide the node ID, but the network number is optional. However, you must provide both the network number and node ID for an uncataloged node on a remote network.

1.1.3.2 Node Names

A node name has the format:

```
//node_name
```

You can use a node name as a node specification only if the node is cataloged (in either your local cache or the `ns_helper` database.) When you use a node name, Aegis gets the internet address associated with that name. If a node is not cataloged, you must use an internet address to specify the node.

Note that you can catalog and name both disked and diskless nodes.

1.1.3.3 Node Specification Examples

The following examples illustrate ways you can specify a node with an ID of A105, a name of `//casey`, and a network number of 4051237A. (These examples assume that `//casey` is cataloged in the `ns_helper` database.)

```
$ lusr -n 0A105
```

Note that hex IDs that start with a letter must be preceded by a '0' for the shell to parse them correctly.

```
$ lusr -n //casey
```

```
$ lusr -n 4051237A.A105
```

If you are using a node on ring 4051237A, you can also use the following internet address to refer to `//casey`:

```
$ lusr -n 0.A105
```

1.2 Using Special Characters

The shell recognizes a variety of special characters that allow you to change the action of commands. The characters in Table 1-1 have special meanings when they appear on a command line. Note that while some of these characters have been discussed as having special meanings in Display Manager (DM) commands, regular expressions, and so forth, those meanings do not necessarily carry over to the Aegis environment. Please be careful to keep the different meanings distinct: for example, you should enclose regular expressions appearing in Aegis commands in quotation marks to avoid confusion.

The at sign (@) is the shell's escape character. You can place an "@" anywhere on the command line to suppress the special meaning of the next character (including the "@" character itself). See *Using Your Aegis Environment* for a full discussion of the usage of shell special characters.

Table 1-1. Command Shell Special Characters

Pathname Wildcards	
Character	Usage
?	Match any single character except newline.
%	Match zero or more characters up to but not including the period.
*	Match zero or more occurrences of the preceding character.
[string]	Match any single character in the character class string.
[^string]	Match any character except those in string.
...	Match zero or more subordinate directories.
=	Copy (derive) leafname from previous argument.
(names)	Group pathnames for use in later derived names.
{expr}	Tag expression for later use.

(Continued)

Table 1-1 Command Shell Special Characters (Cont.)

Input/Output Control		
Character	Usage	Notes
<	Redirect standard input	(3)
<?	Redirect error input	(3)
<</	Read in-line data from standard input	(3)
<<?/	Read in-line data from error input	(3)
>	Redirect standard output	(3)
>?	Redirect error output	(3)
>>	Append standard output	(3)
>>?	Append error output	(3)
	Pipe standard output	(1)
()	Group commands for I/O redirection	(1)
Parsing Operators		
Character	Usage	Notes
#	Comment line in a command file	(4)
&	Run a program or command in the background	(1)
^	Insert parameter	(3)
!	Insert parameter and rescan	(3)
^"cmd"	Insert output of "cmd", with expansion	(3)
^'cmd'	Insert output of "cmd", no expansion	(3)
;	Separate commands on a line	(1)
"	Quoted string, with expansion	(4)
'	Quoted string, no expansion	(4)
@	Escape character	(5)
	Space	(2)

Notes:

1. Special anywhere; causes a new command to start.
2. Special anywhere; causes a new argument to start.
3. Special anywhere; does not start a new argument.
4. Special only at the beginning of an argument.
5. Special only immediately before an otherwise special character.

1.3 The Command Line Parser

Many shell commands that we supply share a standard command line parsing procedure that determines how each command processes command line information. Chapter 2, Shell Commands, and the online help files identify commands that use the command line parser. These commands support the following features:

1. You can use wildcards to specify existing pathnames.
2. You can use derived names to specify logically-related pathnames, and parentheses to create several derived names with one command line. (See Table 1-1)
3. When pertinent, you can include multiple pathnames as command line arguments. For example, `prf file1 file2 file3`.
4. You can use the asterisk character (*) to cause commands to read pathnames from standard input or from another file. For example,
`$ prf */fred/names_file`

prints the files listed in `/fred/names_file`. Also,

```
$ prf *
file1
file2
file3
***EOF***
```

```
$
```

reads the names `file1`, `file2`, and `file3` from standard input, and prints each file. When using the keyboard for standard input, a newline and an end-of-file character must follow the last name. By default, CTRL/Z generates an end-of-file character.

If you include more than one name on an input line, in standard input or in a names file, the command interprets all names except the first one on each line as derived names. For example,

```
$ chn *
a a.old
b b.old
c c.old
***EOF***
```

```
$
```

is equivalent to

```
$ chn * =.old
a
b
c
**EOF***

$
```

Do not confuse the action of the "*" character with that of the input redirection symbol "<". The "*" character causes a shell command to read pathnames from standard input or from another file. The "<" symbol causes a shell command to read data from a file.

1.3.1 Standard Command Options

All shell commands that we supply support the following standard options:

- help** Display detailed usage information.
- usage** Display brief usage summary.
- version** Display software version number.

NOTE: Using any of these three standard options precludes using any other options within the same command.

1.3.2 Command Line Parser Options

Commands that use the command line parser also support the following options:

- ae** Abort if a name in pathname cannot be found. If omitted, processing continues to the next name.
- nq** Do not issue query to verify wildcard names.
- qw** Issue query to verify wildcard names.
- qa** Issue query to verify all names.
- (hyphen alone) Read further data from standard input. End input with CTRL/Z.
- *[pathname]** Read file specified for further pathname arguments. If *pathname* is omitted, read standard input for further pathname arguments.

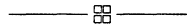
Commands that delete or modify objects automatically verify names specified with wildcards. You can suppress this query using **-nq**, or extend it to all names using **-qa**.

When you select a query option, the command writes the selected names to the error output stream with a ? to prompt you for a response. Then it reads your response from the error input stream (normally the keyboard).

If you respond: The command:

h	Displays help information.
y	Operates on the name.
n	Ignores the name.
q	Quits immediately.
g	Operates on the name and suppresses further name queries.
d <i>new_default</i>	Resets the default. The shell performs the default action when it receives a null line query response (that is, when you simply press RETURN). To change the default, enter d followed by "yes", "no", or "none". The initial default is "none", which means that the command ignores null line responses, and requires explicit yes or no responses.

Chapter 2 describes each shell command in detail. Those commands that use the command line parser refer you to this section for information on the standard options to avoid repetition in the text.



Chapter 2

Aegis Commands

NAME

abtsev – set or display the abort-severity level

SYNOPSIS

abtsev [*options*]

DESCRIPTION

The **abtsev** command lets you set the severity level at which a shell command or program aborts. The abort-severity level is initially set to `-error` when a shell is created. If any command returns a severity level greater than or equal to the abort-severity level, then that shell program, and all its ancestors, are immediately terminated.

The abort-severity level is on a per-shell basis. A new level is established every time a shell program or a new shell is invoked. A shell program inherits the abort-severity level of the preceding level. The severity level is restored when you exit from the shell program.

abtsev is an internal shell command.

See the `pgm_$set_severity` description in the *Domain/OS System Call Reference* for further information on severity levels.

Every shell command or program returns a completion status message to its caller. The message may indicate that the program completed successfully, or it may inform its caller of a fatal internal error. Completion status messages vary in their severity. The following completion status messages appear in order of their severity:

<code>ok</code>	Program completed successfully and performed the requested action.
<code>true</code>	Program completed successfully; its purpose was to test a condition, and the value of that condition was true.
<code>false</code>	Program completed successfully; its purpose was to test a condition, and the value of that condition was false.
<code>warning</code>	Program completed successfully and performed the requested action. However, an unusual (but nonfatal) condition was detected.
<code>error</code>	Program could not perform the requested action because of an error in the input. The output, however, is sound.
<code>output invalid</code>	Program could not perform the requested action because of a syntactic error in the input, and the output is not structurally sound.

internal fatal Program detected an internal fatal error and stopped. The state of the output is unknown.

OPTIONS

Specifying **abtsev** without options displays the current abort-severity level.

-f[false] Set level to false.
-w[arning] Set level to warning.
-e[rror] Set level to error.
-o[utinv] Set level to output invalid.
-i[ntfatal] Set level to internal fatal error.
-p[gmflt] Set level to program fatal error.
-m[ax_severity] Set level to maximum severity error.

EXAMPLES

Show initial setting.

```
$ abtsev
error
$
```

Set level to warning.

```
$ abtsev -w
```

Show new level.

```
$ abtsev
warning
$
```

NAME

acl – list or copy an ACL

SYNOPSIS

acl [*target_object* [*source_object*]] [*options*]

DESCRIPTION

Every directory and file has an associated access control list (ACL) that lists users and their rights to the object. **acl** lets you copy an ACL from one object to another, or display an ACL. For a detailed discussion of ACL structure and usage, please refer to help **edacl**.

ARGUMENTS

target_object (optional) Specify the object whose ACL you want to set or display. You may use a wildcard to specify this argument. Do not, however, specify \$ **acl** /*... (anything) because this may render your node unusable. This wildcard sequence includes files in the /sys tree, which require special ACL settings in order for system software to run.

Default if omitted: use current working directory.

source_object (optional) Specify the file or directory whose ACL(s) is to be used to set the ACL(s) of the target object(s).

Default if omitted: display *target_object*'s ACL

OPTIONS

The following options confine the **acl** command's operation to target objects of the given type.

-d Set or display ACLs of only those target objects that are directories. If used with **-i**, **-id**, or **-if** options, set or display initial ACLs for subdirectories.

-f Set or display ACLs of only those target objects that are files.

The following options control the **acl** command's effect on target objects. If the target object is a directory, they cause **acl** to operate only on the initial ACLs stored within that directory for use on newly created objects, not on the ACL of the directory itself. Note that this does not imply that all the target object(s) are directories. (That is what **-d** specifies.)

-i Set or display initial ACLs. If you are setting the ACLs of a target directory, the source object's type (file or directory) **-i** determines which initial ACL (the one for files or the one for directories) of the target directory is set.

If the target object is a file (or if a wildcard target list includes files) and the source is a directory, you get an error unless you also specify `-is` (so that the initial file ACL in the source directory, rather than the ACL of the directory itself, can be copied to target files). If both source and target are files, then the source file's ACL is applied to the target file, as you would expect. You must run `salld (salvage_directory)` on target directories that have never contained initial ACLs (that is, those directories created using software prior to SR4.1).

- `-id` Set or display only the initial ACLs inside those target objects that are directories that apply to new subdirectories created in those directories.
- `-if` Set or display only the initial ACLs inside those target objects that are directories that apply to new files created in those directories. (Specifying both `-id` and `-if` is the same as `-i`. Specifying neither implies `-d`.)

The following option specifies that one (or both) of the initial ACLs inside the source object is to be copied to the target, rather than the ACL of the source itself. This assumes that the source object is a directory, not a file, since files cannot contain initial ACLs for subordinate objects.

- `-is` Copy the initial ACL(s) in the source object (which must be a directory) to the target. If there is a single target object (either a file or a directory), then the appropriate initial ACL inside the source is applied to the target. If the `-i` option is also specified, then both initial ACLs in the source are copied to the initial ACLs inside those target objects that are directories.

The following option specifies that all the ACLs of the target object(s) are to be set or displayed.

- `-all` Set or display all ACLs of the target object(s). If you are using wildcards to specify the target, you may qualify this action by also specifying `-d` or `-f`. If the source object is a directory, then all of its ACLs (both its own and the two initial ACLs that it applies to newly created subordinate objects) are used to set the corresponding ACLs of the target object(s). If `-is` is also specified, however, the ACL of the source object itself is not used, although all three ACLs of the target directories are still set. Thus you can use `-all` (with or without `-is`) to propagate new ACLs throughout subtrees.

The following options perform miscellaneous tasks:

- `-links` Operate on the links if the *target_object* is a wildcard that specifies link(s). By default `acl` does not operate on links specified with wildcards. However, `acl` always operates on links you specify explicitly (without wildcards). This option does not apply to UNIX hard links, which are always operated on since they are indistinguishable from the original directory entry.

- l** List object names as the command sets ACLs.
- br** Display ACLs only, not object names. `acl` uses the command-line parser, and so also accepts the standard command options listed in `help cl`

EXAMPLES

Assign `old_file`'s ACL to `new_file`.

```
$ acl new_file old_file
```

Set the initial ACLs inside `joe` using the initial ACLs inside `mary` (which must be a directory).

```
$ acl joe mary -i -is
```

Set the initial file ACL in all subdirectories of the current working directory whose names begin with `abc` to the ACL of `file1`.

```
$ acl abc?* file1 -d -if
```

Set the ACLs of all files in the current working directory whose names begin with `abc` to the initial file ACL inside `dir2`.

```
$ acl abc?* dir2 -f -is
```

Set the initial ACLs in all subdirectories of the current working directory whose names begin with `abc`, using the initial ACLs in `dir2`, and the ACLs of all files whose names begin with `abc`, using the initial file ACL in `dir2`. (Adding `-d` confines the operation to directories.)

```
$ acl abc?* dir2 -i -is
```

Set the ACLs of all files matched, using the initial file ACL in `dir2`. The ACLs of all directories matched using the ACL of `dir2` itself. The initial ACLs inside those matched directories are set using the initial ACLs inside `dir2`.

```
$ acl abc?* dir2 -all
```

Set the ACLs of all files matched using the initial file ACL in `dir2`. The ACLs of all directories matched using the initial directory ACL in `dir2`. The initial ACLs inside those matched directories using the initial ACLs inside `dir2`.

```
$ acl abc?* dir2 -all -is
```

SEE ALSO

More information is available. Type

- help acls** For a list of ACL-related commands
- help protection** For general information on Domain protection mechanisms
- help protection acls** For detailed information on ACL structure and usage
- help protection sids** For information on subject identifiers
- help protection rights** For information on access rights

NAME

aqdev – acquire control of a PBU device

SYNOPSIS

aqdev *pathname* [-d[*b*]] [-c *program arg1 arg2 ...*]

DESCRIPTION

aqdev acquires control of a peripheral bus unit (PBU) device. **aqdev** creates a new shell level in which the PBU device driver runs. Release the device by closing this shell level (i.e., type CTRL/Z).

NOTE

This command is valid only if our General Purpose Input/Output (GPIO) software package is installed on your network. See the *Writing Device Drivers with GPIO Calls* for details.

ARGUMENTS

pathname (required) Specify the Device Descriptor File (DDF) for the PBU unit device to be acquired. You can create a DDF by using the **crddf** (**create ddf**) command.

OPTIONS

-d[*b*] Specify debug mode. Display addresses of the mapped DDF, library, etc., along with any errors.

-c[*program arg1 arg2 ...*] Specifies a *program* to run after acquiring the device. This program is run instead of the shell. **aqdev** releases the device after the program returns. This option also allows you to use **aqdev** in a shell script.

EXAMPLES

```
$ aqdev /dev/my_dev
Device 0 acquired.
$ (Run your program using the device.)
$ CTRL/Z
*** EOF ***
Device 0 released.
$

$ aqdev /dev/my_dev -c driver_application
Device 0 acquired.
(driver_application runs using the device.)
Device 0 released.
>>>> $
```

NAME

arcf – maintain an archive file

SYNOPSIS

arcf *command* *arcname* [*pathname* ...]

DESCRIPTION

arcf collects sets of files into one large file and maintains that file as an archive. You can extract files from the archive, add new ones, delete or replace old ones, and list data about the contents. Only text files can be archived.

Files to be added to an archive must exist as files with the name given. Files that are extracted from an archive are written to files with the name given. Files that are added to archives can, of course, be archive files themselves. Any number of files can be nested this way. Thus, you can use **arcf** to maintain tree-structured file directories.

NOTE

When you use the update and print commands, the files are updated and printed in the order they appear in the archived file, not in the order listed on the command line.

ARGUMENTS

- arcname* (required) Specify the name of archive file being created or maintained.
- pathname* (optional) Specify the name of file to be added or deleted from the archive. Multiple names are permitted, separated by blanks. Specifying a hyphen as a filename causes further names to be read from standard input, one per line.
- Default if omitted: perform action on all files in the archive (except **-d**, which requires you to give names explicitly).

COMMANDS

- command* (required) Specify the operation to perform on the archive file *arcname*. Follow the command with **o** get verbose output. Possible commands include the following:
- d** Delete the named files from the archive. If you use the **v** option, filenames are displayed on the standard output as they are deleted from the archive.
 - p** Write the contents of the named files on standard output. The **v** option causes the filenames to precede the file.
 - t** Write a table of contents for the archive file. Normally, the table contains only the filename. If the **v** option is used, the table also includes the file's length, type, and date and time of the last change.
 - u** Update the named archive by replacing existing files, or adding new ones at the end. If you do not give a filename, all possible

files in the archive are updated with files of the same name in the current directory. If the archive file does not exist, it is created with the name given. If you use the `v` option, filenames are displayed on standard output as files are written to the new archived file.

- `-x` Extract the named files from archive. Write each to a file with the same name. If the file already exists, the new version replaces the old. If you add the `v` option, filenames are displayed on standard output as files are extracted.
- `v` Request verbose output. This command can follow any of the other commands (see example below), and causes the archiver to print additional information, generally filenames, on standard output. Its specific action for each command has already been described.

EXAMPLES

Update archive file `my_archive` with a new copy of the file stamps, returning verbose output.

```
$ arcf -uv my_archive stamps
stamps
$
```

Report on the contents of the archive.

```
$ arcf -tv my_archive
stamps      330  local  03/02/88  13:53:07
$
```

NAME

args – echo command line arguments

SYNOPSIS

args [**-err**[**out**]] *string* ...

DESCRIPTION

args writes its arguments, one per line, to standard output unless you specify **-err**. Use it to write to files by redirecting standard output into a file with the **>pathname** expression. The **args** command is useful for inserting messages and diagnostics to be reported to the display to shell scripts and for inserting lines of text into files.

ARGUMENTS

string (required) Specify the string of characters to be written. Multiple strings are permitted; separate strings with blanks. Strings are written one per line. To write phrases containing literal blanks, enclose strings in quotation marks.

OPTIONS

-err[**out**] Write the string(s) to error output instead of to standard output. This option is useful for writing to the transcript pad (where error output is usually directed) from an **args** command inside a pipeline, since standard output is then connected to the pipe.

EXAMPLES

```
$ args Hi there
Hi
there
$
```

```
$ args "Hi there" "Mary"
Hi there
Mary
$
```

Write "Hi there, Mary." into the file **my_file** in the current working directory.

```
$ args "Hi there, Mary." >my_file
```

NAME

bind – combine object modules into an executable file

SYNOPSIS

bind *pathname1* ... [*pathnameN*] [option]...

DESCRIPTION

bind combines two or more object modules into one executable object module. It resolves external references to global symbols and combines sections that have the same name. For full details on the binder, see the *Domain/OS Programming Environment Reference* manual.

The command line simply consists of the word **bind**, one or more pathnames, and zero or more options.

The binder uses the object modules stored in *pathname1* through *pathnameN* to create an executable object file. Each *pathname* must be the name of a valid object file or library file. (A compiler creates an object file, and the librarian creates a library file.) You may use wildcards in pathnames. The binder automatically loads all object modules stored in object files, but conditionally loads the object modules stored in library files.

Options modify the binder's actions. Of all the binder's options, **-binary** is the most important. You must use this option to get an executable output object file.

The following is a summary of the **bind** options. See the *Domain/OS Programming Environment Reference* manual for complete descriptions of each option.

OPTIONS

- align *section-name* long** Aligns the named section on a 32-bit boundary at run time.
- align *section-name* quad** Aligns the named section on a 64-bit boundary at run time.
- align *section-name* page** Aligns the named section on an 8,192-bit boundary at run time.
- allkeepmark** Preserves all marks.
- allmark** Marks all global symbols in the input object files that appear after the option on the **bind** command line.
- allocbss** Gathers all uninitialized global data from C programs and allocates them all to a section named **.bss**.
- allres[olved]** Signals a shell severity level of **error** if there are unresolved global symbols at the end of a **bind** command. This option is useful in controlling shell scripts.

- allunmark** (default) Unmarks all global symbols in the input object files that appear after the option on the bind command line.
- bdir** *directory_name* Adds a pathname to the list of directories the binder searches for input object files.
- b[inary]** *pathname* Creates an output object module and stores it at *pathname*.
- end** Signifies end of a command that is spread over several lines.
- entry** *global_symbol* Specifies a nondefault start address.
- exactcase** Makes the binder case-sensitive to all variable names and section names.
- glo[bals]** Writes currently defined global symbols to error output.
- h[elp]** Prints this list of commands.
- incl[ude]** *module-name* Unconditionally loads the named object module from a library file into the output object file.
- incl[ude] -all** Unconditionally loads all object modules from a library file into the output object file.
- inlib** *pathname* Specifies that the object modules in *pathname* are to be "installed" when the output object file is invoked. (This is an alternative to the **-inlib** utility.)
- localsearch** Forces the binder to make another search through a library file if the previous search loaded an object module containing an unresolved external reference.
- looks[ection]** *name* Makes the named section available for sharing with a public section in an installed library.
- looks[ection] -all** Makes all subsequent sections available for sharing with their counterpart public sections in an installed library.
- mak[ers]** Lists the version numbers of the compilers, binders, etc. that were used to create the input object files.
- map** Writes a complete binder map to standard output.
- mark** *global_symbol* Marks the specified global symbol.
- mark -all** Same as **-allmark**.
- marks[ection]** *section_name* Makes *section_name* public. Affects only those object files that are destined to be installed as an installed library.
- marks[ection] -all** Makes all subsequent sections public. Affects only those object files destined to be installed as an installed library.

- merge[bss]** Merges all sections corresponding to C global variables into a single section named "BSS\$". and gathers all initialized global data from C programs, allocating them to a section named .bss.
- mes[sages] (default)** Produces informational messages at the end of a bind command.
- mod[ule] *new_name*** Changes the name of the output object module from the default (that is, the first input object module loaded) to *new_name*.
- msgs (default)** Same as **-messages**.
- multires** Reports errors if multiple resolutions of the same external symbol exist in object module libraries.
- nmsgs** Same as **-nomessages**.
- noexactcase (default)** Makes the binder case-insensitive to all variable and section names.
- noinlib *pathname*** Specifies that the object file(s) in *pathname* are no longer to be "installed" when the program is invoked.
- nolocalsearch (default)** Searches each library file once, then proceeds to search the next input object file.
- nolooks[ection] *name*** Makes the named section unavailable for sharing.
- nolooks[ection] -all (default)** Makes all subsequent data sections unavailable for sharing.
- nomarks[ection] *section_name***
Makes *section_name* private.
- nomarks[ection] -all** Makes all subsequent sections private.
- nomes[sages]** Suppresses informational messages.
- n[o]multires (default)** Omits error reporting when there are multiple possible resolutions in a library.
- nound[efined]** Suppresses the listing of undefined globals.
- q[uit]** Exits from the binder without finishing.
- readonly[section] *section_name***
Changes the read/write attribute of *section_name* to read-only.

- runtime** *type* Specify the system call semantics (for example, *sys5.3* or *bsd4.3*) that the program requires at runtime. This option creates runtime *type* state resource information (SRI) record in the output object module. The default is the environment specified by the **-systype** option.
- sections** Displays a section map.
- set_version** *number* *number* Sets the program version in the map to the specified number.
- sort[ocation]** Sorts global symbols numerically (by position).
- sort[names]** (default) Sorts global symbols alphabetically (by name).
- stacksize** Display stacksize.
- sys[tem]** Makes system globals visible.
- systype** *type* Builds a system static resource information (SRI) record in the output object module which specifies the resolution of systype dependent links. For *type*, you must specify the name of an operating system (*sys5.3* or *bsd4.3*). This option overrides all system information stored in the input object modules. If **-runtime** is not specified, it also creates a runtime static resource record of the same *type*.
- undefined** Suppresses a listing of unresolved external symbols present at the end of a **bind** command line.
- unmark** *global_symbol* Remove a mark from the specified global symbol.
- unmark -all** Same as **-allunmark**.
- unmarks[ection]** *name* Makes *section_name* private. Affects only those object files that are destined to be installed as an installed library.
- unmarks[ection] -all** (default) Makes all subsequent sections private. Affects only those object files that are destined to be installed as an installed library.
- xref** Displays a listing of cross references.
- (hyphen)** Tells the binder that more input will follow on the next line.

EXAMPLES

A simple binder command line. The binder builds an output object file in `my_program` from two input object files.

```
$ bind a.bin b.bin -binary my_program
```

A library file can also serve as an input object file.

```
$ bind a.bin my_library -b my_program
```

The `-map` option causes `bind` to print substantial binder information.

```
$ bind one.bin two.bin three.bin -map -b my_program
```

The command `bind` specified by itself tells `bind` that more input will follow on the next line. Specify a blank line to end the prompting.

```
$ bind
*paul.bin -allmark -b name.bin
*time.bin -unmark date -unmark year
*john.bin -map *
```

Put comments inside braces.

```
$ bind a.bin b.bin {a comment} -b hope
```

NAME

bldt – display time operating system was built

SYNOPSIS

bldt [*options*] [*node_id*]

DESCRIPTION

bldt displays the time at which the running version of Domain/OS was built.

node_id (optional) Display the build time of the node whose network root directory is *pathname*.

Default if omitted: display build time of current node

OPTIONS

-n *node_spec* ... Display build time of specified node[s].

-a Display build time of all nodes.

EXAMPLES

\$ bldt //os

```
**** Node 29C27.4B51 ****  "//os"
Domain/OS kernel(3), revision 10.0, b120.1 April 15, 1988  1:02:54 pm
```

\$ bldt -n //brazil

```
**** Node 29C27.CBB9 ****  "//brazil"
Domain/OS kernel(8), revision 10.0, b117.3 February 9, 1988  8:12:37 am
```

\$ bldt -n CBB9

```
**** Node 29C27.CBB9 ****  "//brazil"
Domain/OS kernel(8), revision 10.0, b117.3 \
      February 9, 1988  8:12:37 am
```

BLDT

Aegis

BLDT

SEE ALSO

More information is available. Type

help node_spec For details about node specification syntax

NAME

boff – deactivate the shell's **-b** flag

SYNOPSIS

boff

DESCRIPTION

boff turns off the shell's **-b** (display output of background process) flag, which is turned on by the **bon** command or the **-b** option on the shell command line. When the flag is off, the output of background processes created with the **&** parsing operator is sent to **/dev/null**. This background process output is sent to **/dev/null** by default.

boff requires no arguments or options.

NAME

bon – activate the shell's **-b** flag

SYNOPSIS

bon

DESCRIPTION

bon activates the shell's **-b** (display output of background process) flag. You can also activate the flag by using the **-b** option on the **sh** command line when the shell is invoked. The **boff** command deactivates the **-b** flag. By default, the flag is off when a shell is invoked.

This flag causes the shell to send the output of a background process (created with the **&** parsing operator) to the display. The output of the background process is displayed in the transcript pad of the shell where it was invoked.

You turn **bon** on in a shell script. It remains on until that shell script exits, or until you override it by a **boff** command in a nested shell script. When a shell script exits, the state of execution tracing is returned to the state in effect just before you invoked the script.

bon requires no arguments or options.

NAME

calendar – set system calendar clock

SYNOPSIS

calendar (from the shell)

ex calendar (from the Mnemonic Debugger)

DESCRIPTION

Use **calendar** to set or reset the calendar clock in a node. You can also use it to update the last valid time known to the system, which is stored on the boot volume. Normally, the clock is set at the factory, and there is no need to reset it. You should take care if you set the clock backwards in time, since duplicate unique identifiers (UIDs) may be generated, resulting in the loss of files. For information on changing the timezone, see the **tz** (timezone) command description.

Note that **calendar** works only on unmounted volumes, so you must invoke it from the Mnemonic Debugger in order to set the clock on the boot volume.

calendar prompts for all required arguments and options.

NAME

catf – read file(s) and write to standard output

SYNOPSIS

catf [*pathname* ...]

DESCRIPTION

catf reads input files in order and writes them to standard output.

ARGUMENTS

pathname (optional) Specify file(s) to write to standard output. If you give multiple *pathnames*, they are read and written in the order that they appear on the command line.

Default if omitted: read standard input

EXAMPLES

\$ catf garbage

Writes the file **garbage** on standard output.

\$ catf garbage – trash >collector

Concatenates the file **garbage**, the lines read from standard input, and the file **trash**, and writes the result in the file **collector**.

\$ catf collector >>junk

Appends the contents of **collector** to the file **junk**.

NAME

chhdir – change a log-in home directory

SYNOPSIS

chhdir *pathname*

DESCRIPTION

The log-in home directory contains your initial working and naming directories. After login, you are automatically in your log-in home directory. Use **chhdir** to change your log-in home directory.

ARGUMENTS

pathname (required) Specify name of new log-in home directory.

EXAMPLES

Set new log-in home directory to `//user/john`.

```
$ chhdir //user/john
```

NAME

chn – change an object's name

SYNOPSIS

chn *old_name* [*new_name*] [*old_name* [*new_name*] ...] [*options*]

DESCRIPTION

chn changes the name of a file, directory, or link. **chn** works with the rightmost component ("leafname") of the old name (see EXAMPLES).

You cannot use **chn** to change the name of a directory embedded in a complete pathname, if doing so would relocate the file to some other part of the naming tree. For instance,

```
$ chn //et/mary/letters //et/fred/letters
```

is illegal. Use the **mvf** (*move_file*) command for that operation.

Multiple *old_name/new_name* pairs and pathname wildcarding are permitted.

ARGUMENTS

old_name (required) Specify the current pathname of the object to be renamed.

new_name (optional) Specify the new name of the object. The new name may be derived from the old name. *new_name* may be omitted entirely if **-d**, **-y**, or **-u** are specified. Otherwise, some portion of it is required. Names may be 1 to 32 characters long.

Default if omitted: derive *new_name* from *old_name*

OPTIONS

-p[*airwise*]

Instructs **chn** to accept the list of tokens as consecutive pairs. This is necessary whether you are using wildcards or not.

-d Append today's date (month and day) to *new_name* in the form *new_name.mm.dd*

-y Append today's date (year, month, and day) to *new_name* in the form *new_name.yy.mm.dd*

-u Force *new_name* to be unique by appending a sequence number to the end of the name until it becomes unique.

-s List names changed on standard output.

NOTES

If you use more than one pair of name tokens with this command, you must use the `-p` option. It instructs the command to accept the list of tokens as consecutive pairs. This is necessary whether you are using wildcards or not. In the past, this command has correctly paired off tokens without the prompting of a switch; now the `-p` switch must be used to achieve this result. The switch has been added to protect against inadvertent use in a shell, other than `/com/sh`, where wildcard expansion can be a problem.

EXAMPLES

Change the name `fritz` to `henri` in the current working directory.

```
$ chn fritz henri
```

Change `henri` to `mike` and `peter` to `paul`.

```
$ chn henri mike peter paul
```

Change `a`, `b`, and `c` to `a.zorp`, `b.zorp`, and `c.zorp`.

```
$ chn (a b c) =.zorp
```

Change the file `projects` to `new_projects` in the directory `/my/stuff`.

```
$ chn /my/stuff/projects new_projects
```

Change `henri` to `henri.mm.dd` where `mm` is the current month (01-12) and `dd` is the current date (01-31).

```
$ chn henri -d
henri.07.19
```

Change `joe` by appending sequence number to end of filename.

```
$ chn joe -u
joe.1
```

NAME

chpass – change a log-in password

SYNOPSIS

chpass [*new_password*]

DESCRIPTION

chpass changes your log-in password to *new_password*. **chpass** allows you to change your password from the shell command level.

ARGUMENTS

new_password (optional) Specify new password. Omitting this option causes **chpass** to prompt for your new password. Input echo is disabled. A second prompt verifies the password and guards against typing mistakes.

Default if omitted: prompt up to three times for password

EXAMPLES

Set new log-in password to sesame.

```
$ chpass sesame
```

NAME

chpat – replace pattern in text file

SYNOPSIS

chpat [*options*] [*pathname ...*] [-p] [*pat ...*]
from_pattern [*to_expression*]

DESCRIPTION

chpat copies every line from its input files to its output files, globally substituting the text replacement pattern *to_expression* for each occurrence of *from_pattern* in those lines designated by the *pat* argument(s) and any options.

Refer to the descriptions of the **ed** (*edit*), **fp** (*find_pattern*), and **edstr** (*edit_stream*) commands for related information.

ARGUMENTS

pathname (optional) Specify name of file to be searched. Multiple pathnames and wildcarding are permitted.

Default if omitted: search standard input

from_pattern (required) Specify target text string (a regular expression) for substitution or deletion. If the string includes the characters % \$ [] { } ! * or any other shell special characters, enclose it in quotes to avoid unpredictable results. If the *pathname* argument is present, precede this argument (or the *pat* argument, if present) with -p to separate the pathname(s) from the regular expressions on the command line.

to_expression (optional) Specify replacement string. If you do not specify a replacement, the *from_pattern* is deleted. If regular expressions defining a range of text (*pat* argument) are present, you must use a literally null *to_expression* ("") to delete *from_pattern*

pat (optional) Specify range of text for which the substitution is to apply, in the form of a regular expression. Multiple expressions separated by blanks are permitted. Unless modified by options, any line of text matching any pattern is replaced and all lines, changed or not, are written to output. If the *pathname* argument is present, precede this argument with -p to separate the pathname(s) from the regular expressions on the command line.

Default if omitted: use *from_pattern* to select matching lines

OPTIONS

- a** Select only lines that match all the leading expressions, in any order.
- x** Select only lines that match none of the leading expressions.
- o** Write only the selected lines to standard output. By default, **chpat** writes all lines to output.
- l** List name(s) of input file(s) on output file(s) as the input file(s) are searched.
- out *pathname*** Specify name of output file. Note that this option is position dependent and must follow the input *pathname* if you specify both. Pathname may be derived from the input filename. If this option is omitted, output is written to standard output.

EXAMPLES

Changes all occurrences of **foo** in standard input to **bar** and writes the results to standard output.

```
$ chpat foo bar
```

In lines starting with **This**, it changes all occurrences of multiple spaces to a single space.

```
$ chpat '%This' " *" ' '
```

Works on lines starting with either **This** or **That**.

```
$ chpat '%This' '%That' " *" ' '
```

In lines that start with **when** and end with **only**, change all semicolons to colons.

```
$ chpat -a '%when' 'only$' ';' ':'
```

In lines that do not contain either **not** or **none**, change all instances of **some** to **all**.

```
$ chpat -x not none some all
```


Delete (replace with nothing) all occurrences of erase.

```
$ chpat erase
```

Exactly the same effect can be obtained with

```
$ chpat erase ''
```

Change all occurrences of the string *if x = y* to *if (x = y)* in all Pascal source files (files ending with .pas) and put the output for x.pas in x.pas.new.

```
$ chpat ?*.pas -out =.new -p "if x = y" "if (x = y)"
```

SEE ALSO

More information is available. Type

help patterns For details about regular expression syntax and usage

NAME

cmf – identify differences among files

SYNOPSIS

cmf *file_a* [...*file_e*] [*options*]

DESCRIPTION

cmf compares the contents of two to five ASCII files and reports the differences on standard output. This command works only on files: to compare directory trees, use **cmt** (**compare_tree**).

ARGUMENTS

file_a (required) Specify pathname of original file; all differences are reported in relation to this file. Wildcarding of this pathname is permitted to achieve multiple comparisons.

file_b ... *file_e* (optional) Specify descendants of *file_a*. If more than one file is specified, you may use a hyphen (-) to cause standard input to be read in place of a pathname.

Default if omitted: read standard input

OPTIONS

-r *pathname* Report all differences to the specified report file, in addition to reporting to standard output. This pathname may be derived from *file_a* (if *file_a* is wildcarded) to produce one report for each comparison. If *file_a* is wildcarded and this report filename is not derived, all reports are concatenated into the single report file.

-tb Include trailing blanks in the comparison. By default, ignore trailing blanks. **-tb** also causes **cmf** to regard the newline character at the end of the last line in the file as significant (if it exists).

-br Display only line numbers of lines containing discrepancies. By default, display both line numbers and line contents.

-l Display names of files being compared before each comparison is performed. This is useful when wildcarded pathnames are specified.

-m *n* Set the minimum number of lines for a rematch to *n*. This is the minimum number of lines, following a reported difference, that must match for **cmf** to consider the files synchronized. The default value is 3.

EXAMPLES

Assume that **file1** contains

```
The large  
brown fox jumped  
over the fence.
```

and **file2** contains

```
The large  
blue moon  
in the  
morning sky  
was visible  
over the fence.
```

cmf produces the following output when **file1** is compared to **file2**.

```
$ cmf file1 file2
```

```
A2      brown fox jumped  
changed to  
B2      blue moon  
B3      in the  
B4      morning sky  
B5      was visible
```

```
1 discrepancy found
```

SEE ALSO

More information is available. Type

help cmsrf For details about comparing sorted files

help cmt For details about comparing directory trees

NAME

cmsrf – find lines common to two files

SYNOPSIS

cmsrf [*options*] *file1* [*file2*]

DESCRIPTION

cmsrf reads sorted files, *file1* and *file2*, and produces 1-, 2-, or 3-column output. Column 1 contains lines found only in *file1*, column 2 contains lines found only in *file2*, and column 3 contains lines found in both files. The number option, *-n*, specifies which columns to print. Use *cmf* (compare_file) to compare unsorted files.

Use of a hyphen for either filename causes the data to be read from standard input.

If no options are specified, **cmsrf** produces a complete 3-column report.

ARGUMENTS

Use of a hyphen for either filename causes the data to be read from standard input.

file1 (required) Specify first file for comparison.
file2 (optional) Specify second file for comparison.

Default if omitted: compare *file1* to standard input

OPTIONS

If no options are specified, **cmsrf** produces a complete 3-column report.

-n Specify number(s), where *n* is an integer sequence representing the following:

- 1 Report only lines exclusive to *file1*.
- 2 Report only lines exclusive to *file2*.
- 3 Report only lines common to both files.

EXAMPLES

Compare *//us/sorted_stuff.c* to standard input and report lines found in either place, but not both.

```
$ cmsrf -12 //us/sorted_stuff.c
```

Report only common lines for both files.

```
$ cmsrf -3 //us/sorted_stuff.a //us/sorted_stuff.b
```

SEE ALSO

More information is available. Type

help cmf For details about comparing unsorted files

help cmt For details about comparing directory trees

NAME

cmt – compare source tree to target tree

SYNOPSIS

cmt *source_pathname target_pathname [options]*

DESCRIPTION

cmt compares all the objects in the source tree against all objects in the target tree. **cmt** reports any objects cataloged in the source that do not also appear in the target. If the target contains objects that do not appear in the source, however, the differences are ignored.

cmt compares objects based on their internal representation, unlike **cmf** (**compare_file**), which treats its input data as ASCII text streams and compares them as such.

Both the source and target pathnames must specify the same type of object, either a directory or a file. However, **cmt**, can compare objects of any type, unlike **cmf**, which compares only text files.

If **cmt** encounters differences, it reports that the objects are different and continues the comparison with the next object.

ARGUMENTS

Use of wildcards in pathnames is permitted. Multiple source/target pairs are permitted.

source_pathname (required) Specify source tree.

target_pathname (required) Specify target tree. Name may be derived from *source_pathname*.

OPTIONS

If no options are specified, **cmt** reports only the names of directories and files with differences in source and target trees.

- l** List all directories and files compared.
- ld** List all directories compared.
- lf** List all files compared.
- ae** Abort on the first mismatch, or if the source tree contains a name not found in the target tree. By default, the comparison continues after the mismatch is reported.

EXAMPLES

Assume that the directories **dir1** and **dir2** each contain three files called **a**, **b**, and **c** and that the contents of the **b** files differ. The following is the result of a comparison of those two directories.

```
$ cmt dir1 dir2
*** compare failed at file loc 0 SRC: 10002 DST: 100011
dir1/b - compare failed (from US / file utility)
```

SEE ALSO

More information is available. Type

help cmf	For details about comparing unsorted files
help cmsrf	For details about comparing sorted file

NAME

cpboot – copy the system boot file `sysboot`

SYNOPSIS

/etc/cpboot source_directory target_directory

DESCRIPTION

cpboot copies the system boot file `sysboot` from one directory to another. The `sysboot` file is used by the bootstrap prom to start the system. **cpboot** is useful for copying `sysboot` to a floppy disk, thus making the standalone utilities (`sau`) directory on the floppy disk accessible from the boot prom. You may also use it to update a Winchester disk when a new software release is distributed.

source_directory (required) Specify directory containing the file `sysboot`.

target_directory (required) Specify directory to which `sysboot` is to be copied. This must be the entry directory on the target logical volume.

NAME

cpf – copy a file

SYNOPSIS

cpf *source_pathname* [*target_pathname*] [*options*]

DESCRIPTION

cpf copies a file from the source pathname to the target pathname. **cpf** copies only files; see **cpt** (*copy_tree*) for copying directories and their subordinate objects.

Multiple source/target pairs and pathname wildcarding are permitted.

ARGUMENTS

source_pathname (required)

Specify file to be copied. If the source pathname is a link name, **cpf** resolves the link and copies the file to which the link refers.

target_pathname (optional)

Specify target for copy. If *target_pathname* is a directory, *source_pathname* is copied into this directory. The target must not be a link.

Default if omitted: copy *source_pathname* into current working directory

Multiple source/target pairs and pathname wildcarding are permitted.

OPTIONS

- p**[*airwise*] Instructs **cpf** to accept the list of tokens as consecutive pairs. This is necessary whether you are using wildcards or not.
- c** (default) Create source file at target. An error occurs if the target file already exists.
- r** Replace target with copy of source.
- lf** List files copied.
- ldl** List files deleted as a result of a **replace** (**-r**).

-chn	Use with -c to change the name of an existing object with the target pathname before the copy is made. Use with -r to change the name of a target file if it is in use and cannot be deleted.
-dacl (default)	Apply the target directory's default ACL for files copied. In addition to its own ACL, each directory has two default ACLs, one for its files and another for its subdirectories. The system assigns the parent directory's default ACL for files to the target file unless you specify -sacl or the file belongs to a protected subsystem (see the -subs option).
-sacl	Retain the source file's ACL.
-subs (default)	Retain source ACL for objects that belong to subsystems.
-nsubs	Apply the target directory's default ACL for objects that belong to subsystems.
-f	Force deletion of target object if 'p' rights are present.
-du	Delete when unlocked. This option is useful with -r . If the object to be replaced is locked when cpf is invoked, the replace operation is performed when the object is unlocked.
-pdt	Preserve the source file's modification and used times.
-cwl	Obtain a co-writer's instead of an exclusive lock on files being copied.

NOTES

If you use more than one pair of name tokens with this command, you must use the **-p** option. It instructs the command to accept the list of tokens as consecutive pairs. This is necessary whether you are using wildcards or not. In the past, this command has correctly paired off tokens without the prompting of a switch; now the **-p** switch must be used to achieve this result. The switch has been added to protect against inadvertent use in a shell, other than **/com/sh**, where wildcard expansion can be a problem.

EXAMPLES

Copy the file **wbak** from the **latest** directory to the current directory, and call it **wbak.latest**

```
$ cpf /latest/wbak wbak.latest
$
```

Copy the file **latest/com/wbak** to the **com** directory, replacing the existing **com/wbak**

```
$ cpf /latest/com/wbak /com -r
$
```

Copy and list all files in the games directory starting with space, to the working directory.

```
$ cpf /games/space?* -lf
(file) "space_war" copied.
(file) "space_walk" copied.
(file) "space_shot" copied.
$
```

Copy all files in the working directory with the suffix .pas to the directory backup, and append a date.

```
$ cpf ?*.pas backup/=.12.07
$
```

NAME

cpl – copy a link

SYNOPSIS

cpl *linkname* [*pathname*] ... [*options*]

DESCRIPTION

cpl copies a linkname to the target object.

Multiple linkname/pathname pairs and wildcarding are permitted.

ARGUMENTS

linkname (required) Specify the name of the link to be copied.

pathname (optional) Specify the target *pathname* of the copied link. If *pathname* is a linkname, then this link is created or replaced (depending on various options below). If *pathname* is a directory, the link text is copied into this directory. In no case is the object to which the link refers affected; only the text of the link itself.

Default if omitted: copy link into current working directory

OPTIONS

-c (default) Create source link at target. An error occurs if the target link already exists.

-r Replace target with copy of source.

-ll List links copied.

-ldl List links deleted because of replacement (**-r**).

-chn Change name of existing link with *target_pathname* before copying.

-p[airwise] If you use more than one pair of name tokens with this command, you must use the **-p** option. It instructs the command to accept the list of tokens as consecutive pairs. This is necessary whether you are using wildcards or not. In the past, this command has correctly paired off tokens without the prompting of a switch; now the **-p** switch must be used to achieve this result. The switch has been added to protect against inadvertent use in a shell, other than /com/sh, where wildcard expansion can be a problem.

EXAMPLES

Copy the link //ai/sources to the node entry directory as **progs**.

```
$ cpl //ai/sources /progs
```

Copy the link `/sys/print` from the node whose entry directory is `zorba` to the local `/sys` directory, replacing any existing link.

```
$ cpl //zorba/sys/print /sys -r
```

NAME

cpscr – copy the current display to a file

SYNOPSIS

cpscr [**-inv**] [**-append**] [**-gpr[_bitmap]**] *pathname*

DESCRIPTION

cpscr copies the current screen image, without clearing it, to the file you specify. Use the **prf** (**print_file**) command to print the file.

Use the DM command **cpo** to copy the screen without creating a new process window or changing the current transcript pad. **cpo** invokes the **cpscr** command from the DM without creating a pad or window. Thus, press <CMD> and type

```
cpo /usr/apollo/bin/cpscr pathname
```

You may copy small portions of a black and white screen (such as a single window) with the DM command **xi**.

By default, black and white screens are copied into a GMF file. Color screens are copied into a GPR bitmap.

pathname (required) Specify file to that the screen is copied.

OPTIONS

- inv** Invert image. Use this option to store the image in reverse video. Black screen pixels become white and white screen pixels become black. Do not use this option with the **-gpr_bitmap** option or on color nodes.
- append** Appends a black and white screen image to an existing GMF file. You cannot use this option with the **-gpr_bitmap** option or on color nodes.
- gpr_bitmap** Use this option to copy a black and white screen into a GPR bitmap file rather than a GMF file. This option has no meaning for color nodes since color screens are already copied into GPR bitmaps.

EXAMPLES

Invert and copy the current screen image to the specified file. Since the command line is echoed in the shell's process transcript pad prior to execution, this command will appear in the resulting image.

```
$ cpscr -inv //us/looky_there
```

<cmd>

Command: **cpo /usr/apollo/bin/cpscr -inv //us/looky_there**

Same result as in the previous example, but the cpscr line will not appear in the plotted output.

SEE ALSO

More information is available. Type

- | | |
|-----------------|--|
| help xi | For details about copying small portions of the screen |
| help prf | For details about printing the screen copy file |
| help cpo | For details about the DM command cpo |

NAME

cpt – copy a directory tree

SYNOPSIS

cpt *source_pathname target_pathname ... [options]*

DESCRIPTION

cpt is a multipurpose tool for copying, merging, and replacing files, directories, and links. To copy files only, use **cpf** (**copy_file**).

ARGUMENTS

Multiple source/target pairs and wildcarding are permitted.

source_pathname (required)

Specify the file, link, or directory tree to be copied. **cpt** does not change the contents or link references of the source, so errors leave the source unaffected.

target_pathname (required)

Specify the file or directory tree to be created, replaced, or merged. *target_pathname* may be derived from the source path-name. The target cannot be a link. In addition, the target cannot be a logical volume entry directory, or the network root unless the **-md** option is specified.

OPTIONS

- p[airwise]** Instructs **cpt** to accept the list of tokens as consecutive pairs. This is necessary whether you are using wildcards or not.
- af date** Copy only objects whose DTMs (date-times) are after the given date and time: [[*yyyy/mm/dd*][.*hh:mm:ss*]] | today. The date defaults to today, and the time to midnight, if either is omitted from *date*.
- be date** Copy only objects whose DTMs are before the given date and time: [[*yyyy/mm/dd*][.*hh:mm:ss*]] | today. The date defaults to today, and the time to midnight if either is omitted from *date*.
- c (default)** Create source at target. If the file or directory already exists, an error occurs and processing continues to the next source/target pair. Not valid if **-ms**, **-md**, or **-r** is specified.

If the source is a file, **cpt** copies it to the target. If the source is a directory, **cpt** copies the directory to the target. It then copies every file cataloged in the directory (and all subdirectories) until it reaches the end of the tree.

Each link name in the source tree is created as a link name in the

target, but the object that the link references is not copied. If *source_pathname* is itself a link, however, the link is resolved and the object to which it points is copied to the target.

- r** Replace target with source. Not valid if **-c**, **-ms**, or **-md** is specified. **cpt** deletes the tree starting at the target pathname and copies the entire source tree in its place. Note that the target is deleted before copying begins. If no target tree by the specified name exists, **cpt** creates one and duplicates the source.
 - ms** Merge source and target if both are directories. Not valid if **-c** or **-r** is specified. If the target does not exist, **cpt** duplicates the source at the target. If the target exists, **cpt** merges the source into the target, replacing files and links, and combining directories.
- If both the source and the target are directories, **cpt** merges their contents as described below. Otherwise, **cpt** deletes the target and replaces it with the source.
- To merge directories, **cpt** compares their contents, object by object. Objects that exist in the source but not in the target are created in the target. Objects that exist in the target but not in the source remain unchanged. Files and links with the same name in both the source and the target are deleted from the target and replaced by the source version. Directories with the same name in both source and target are merged. **cpt** continues this process recursively until it reaches the end of the source tree.
- md** Merge source and target if both are directories. Similar to **-ms** except that files and links with the same name in both source and target are left unchanged in the target.
 - f** Force deletion of target object if 'p' rights are present.
 - dacl** (default) Apply the target directory's default ACLs. In addition to its own ACL, each directory has two default ACLs, one for its files and another for its subdirectories. **-dacl** causes **cpt** to apply the target directory's default ACLs to each subdirectory and file it copies. The **-sacl** option causes each object to retain its original ACL.
 - sacl** Retain the source ACL.
 - chn** Use with **-c** to change the name of a target before source is copied. Use if *target_name* already exists. Use with **-r**, **-ms**, and **-md** to change the *target_name* if target is in use.

-subs (default)	Retain source ACL for objects that belong to subsystems.
-nsubs	Apply the target directory's default ACL for objects that belong to subsystems.
-pr <i>pathname</i>	Preserve the object <i>pathname</i> in the target when another object with the same name exists in the source. Valid with -ms option only.
-pdt	Preserve the source's modification and used times.
-cwl	Obtain a co-writer's instead of an exclusive lock on files being copied.

The following five options allow you to monitor **cpt**'s operation. You can use **-ld**, **-lf**, and **-ll** in any combination. By default, the listing options apply to both copied and deleted objects. To list only deletions, use **-ldl** with **-l**, **-ld**, **-lf**, or **-ll**.

-l	List all objects as they are copied.
-ld	List directories as they are copied.
-lf	List files as they are copied.
-ll	List links as they are copied.
-ldl	List only objects deleted as a result of replacements.

NOTES

If you use more than one pair of name tokens with this command, you must use the **-p** option. It instructs the command to accept the list of tokens as consecutive pairs. This is necessary whether you are using wildcards or not. In the past, this command has correctly paired off tokens without the prompting of a switch; now the **-p** switch must be used to achieve this result. The switch has been added to protect against inadvertent use in a shell, other than **/com/sh**, where wildcard expansion can be a problem.

Five conditions always terminate execution:

- An attempt to use the network root or node entry directory as a target, without specifying a merge.
- An error in reading the top level of the source tree.
- An attempt to create an existing directory (if the target is an existing directory, you must specify **-r** or **-m**).
- The logical volume containing the target directory is full.
- A quit or stop fault in the process.

EXAMPLES

Copy the directory tree **/com** to **/com.backup** replacing the existing **/com.backup** tree.

```
$ cpt /com /com.backup -r
```

Merge the directory tree `my_circuits` into the `/circuits` tree.

```
$ cpt my_circuits /circuits -ms
```

SEE ALSO

More information is available. Type

help datetime For more information on date-time syntax used with `-af` and `-be`

NAME

crd – create a directory

SYNOPSIS

crd *pathname* ...

DESCRIPTION

crd creates a directory with the specified *pathname*.

ARGUMENTS

pathname (required) Specify the subdirectory name to be created. Multiple pathnames are permitted. The new directory receives its parent directory's initial ACL.

EXAMPLES

Create the subdirectory *new_dir*

```
$ crd /my_dir/new_dir
```

NAME

crddf – create, display, or modify a device descriptor file

SYNOPSIS

crddf [*options ...*] *pathname*

DESCRIPTION

crddf creates, displays, or modifies a device descriptor file (DDF). A DDF defines a peripheral bus unit (PBU) device for which you have written a driver. See *Writing Device Drivers with GPIO Calls* for details on both DDFs and PBUs.

crddf is valid only if the general purpose input/output (GPIO) software is running on your network.

pathname (required) Specify name of the DDF to be created, modified, or displayed.

OPTIONS

- Read further options from standard input. Signal completion with **-end**.
- at** Specifies that device lives on the AT-compatible bus.
- call_library *pathname***
Specify *pathname* of the call side of the device driver library. This option is required.
- check** Check the DDF to ensure that all required fields have been specified.
- cleanup_routine [*entry_name*]**
Specify the entry-point name of the clean-up routine in the call library. Omitting the entry name deletes a previously existing clean-up routine.
- csr_offset *port_number***
Specify the offset into the control status register (CSR) page, in hexadecimal format, at which the device's control/status registers are located. Device drivers may use this information during controller initialization.
- csr_page *iova*** Specify the hexadecimal address of the CSR page for the device in the bus address space. The following information applies to the particular bus structure implemented on your system:
 - Multibus: optional
 - VME bus: optional. If specified, must be page-aligned and in the range C000-D000.

- AT-compatible bus: If specified, may indicate a range (for example, `-csr_page 200 21F`). If the second parameter is missing, a range of 8 consecutive bytes is assumed (for example, `-csr_page 200` assumes a range of 200-207).
- `-debug` Sets a flag that can be used to turn on debugging logic in a driver.
- `-display` Display the current contents of the DDF.
- `-dma_channel channel-number` Specifies to the driver the DMA channel number used by AT-compatible device. This is a Version 3 option.
- `-end` Close the updated DDF and exit.
- `-initialization_routine entry_name (required)` Specify the entry-point name of the initialization routine in the call library.
- `-interrupt_library pathname` Specify the pathname of the interrupt side of the device driver library.
- `-interrupt_routine level [entry_name] (required)` Specify a level at which the device interrupts and the entry-point name of an optional interrupt routine.
- `-major ddevice_number` Specify the DDF's major device number in range 0-31.
- `-minor ddevice_number` Specify the DDF's minor device number in range 0-511.
- `-memory_base iova` Specify the MULTIBUS address that marks the base of a controller's local memory. If the specified *iova* is less than 64K this is a Version 2 option, if *iova* is greater than 64K, this is a Version 3 option.
- `-memory_size length` Specify the size, in hexadecimal format, of the controller memory. If the specified *iova* less than 64K, this is a Version 2 option; if greater than 64K, this is a Version 3 option.
- `-multiple` Specify that the device driver supports more than one device and cause the `crddf` command to check the driver entry-point names listed in the DDF for each device to ensure that it doesn't load multiple copies of the same driver.

- node[f]** [*node number*!*] (required)
Specify the hexadecimal node ID of the node to which the device is physically connected. **-nodef** suppresses the check which makes certain the node exists. You may use an asterisk (*) instead of the node ID to indicate the local node.
- quit**
Exit without modifying the original DDF.
- remddf** //*node name*
Specify a remote node on which the DDF resides.
- replace**
Replace (i.e., overwrite) an existing DDF with a new version. To modify only selected portions of an existing DDF, use **-update**.
- revision** [*string*]
Specify an optional revision number as an 8-character string.
- serial_number** [*string*]
Specify an optional serial number as a 16-character string.
- share**
Specify a DDF for a controller that can be shared among multiple processes.
- stack_size** [*decimal number*]
Specify the number of bytes, in decimal, to be allocated to the interrupt stack (default is 1024).
- type** *type name*
Specify the DDF's type. The type must already be installed on the node.
- unit** *unit number* (required)
Specify the unit number of the device (must be equal to the lowest interrupt level on which the device interrupts).
- MULTIBUS: Must be in range 0-5.
 - VME bus: Must be in range 8-14.
 - AT-compatible bus: Must be in range 0-15.
- update**
Modify selected portions of an existing DDF. If this option is specified, it must precede all other options on the command line. To replace a DDF completely, use **-replace**.
- user_info** [*string*]
Specify up to 64 characters of optional user information (no embedded blanks).
- vme**
Specifies that device lives on VME bus. This is a Version 3 option.
- 20_bit_addressing**
Specify 20-bit memory address size of controller. You must use PBU2 calls.

EXAMPLES

1. Create a new DDF specifying only the required information.

```
$ crddf /dev/mt0 -  
New DDF.  
> -unit 3  
> -node 2F  
> -csr_page 1400  
> -call_library /lib/mt.lib  
> -initialization_routine mt_$init  
> -interrupt_library /lib/mt.int.lib  
> -interrupt_routine 3 mt_$int  
> -check  
No missing fields.  
> -end  
$
```

2. Display a DDF.

```
$ crddf /dev/mt0 -display  
ddf version: 1  
device uid: 00030003 0000002F (unit 3, node 2F)  
csr page iova: 1400  
call library: /lib/mt.lib  
interrupt library: /lib/mt.int.lib  
initialization entry point: mt_$init  
cleanup entry point: mt_$cleanup  
interrupt stack size: 1024  
interrupt routines:  
level 0: [unused]  
level 1: [unused]  
level 2: [unused]  
level 3: mt_$int  
level 4: [unused]  
level 5: [unused]  
level 6: [unused]  
level 7: [unused]  
serial number:  
revision:  
user info:  
$
```


3. Change the name of the interrupt routine in an existing DDF.

```
$ crddf /dev/mt0 -update -interrupt_routine 3 mt_$sio
```

4. Replace a DDF on the node //grip with a new version.

```
$ crddf -remddf //grip /dev/x25 -
> -replace
> -unit 2
> -node *
> -call_library /sys/x25/x25_driver.lib
> -interrupt_library /sys/x25/x25_driver_int.lib
> -initialization_routine x25_driver_$init
> -cleanup_routine x25_driver_$cleanup
> -interrupt_routine 2 x25_driver_$int
> -memory_base 7000
> -memory_size 1000
> -revision 7.0
> -serial_number
> -user_info release
> -display
> -end
$
```

5. Create a new DDF for a device that will be accessed through streams for the installed type foodev:

```
$ crddf /dev/foodev -
New DDF.
> -unit 3
> -node *
> -csr_page 1400
> -call_library /lib/foodev.lib
> -initialization_routine foodev_$init
> -interrupt_library /lib/foodev.int.lib
> -interrupt_routine 3 mt_$int
> -type foodev
> -check
No missing fields.
> -end
$
```

NAME

crefs – cross-reference symbols in a file

SYNOPSIS

crefs [*pathname ...*] [**-k** *keyfile*] [*options*]

DESCRIPTION

crefs produces a cross-referenced list of the symbols in each of the named files, and writes each list to standard output. A symbol is a string of letters, digits, underscores, and dollar signs and must begin with a letter. The list contains every symbol in the file in alphabetical order, followed by the numbers of the lines in which the symbol appears.

Symbols of more than 32 characters are truncated.

ARGUMENTS

pathname (optional) Specify input file. Multiple pathnames and wildcarding are permitted: separate names with blanks.

Default if omitted: read text from standard input

OPTIONS

If you do not specify the **-f** option, **crefs** treats uppercase and lowercase letters as different characters, and places uppercase letters before lowercase letters in the alphabetical sort.

-f Treat all input text as lowercase while cross-referencing.

-k *key_file* Only the words listed in *key_file* are cross-referenced. List these words one per line.

EXAMPLES

To find all occurrences of certain variables in the program *cycle*, type:

```
$ crefs cycle
```

You can also use **crefs** in conjunction with other commands to produce more refined results. For instance:

```
$ crefs cycle | tee cycle.all | fpat wheel spoke axle >cycle.some
```

The output file *cycle.all* contains a list of all the symbols in the program, with references to the line containing them. The output file *cycle.some* contains only the lines with references to the three variables named: *wheel*, *spoke*, and *axle*.

NAME

crf – create a file

SYNOPSIS

crf *pathname*...

DESCRIPTION

crf creates a zero-length file with the specified pathname. The new file receives its parent directory's initial ACL for files.

ARGUMENTS

pathname (required) Specify file to be created. Multiple pathnames are permitted, separated by blanks.

EXAMPLES

\$ crf my_file

NAME

cr1 – create a link

SYNOPSIS

cr1 *linkname* *object_name* ... [-r]

DESCRIPTION

cr1 is used to create links. Links normally serve two functions: as a shorthand way of specifying objects with long (and frequently recurring) pathnames and as static pointers to other objects. Links cause the shell to redirect a pathname to another object. In effect, links allow you to take a detour from one part of the naming tree to another.

Multiple *linkname/pathname* pairs are permitted. Wildcards are not permitted with *linkname/pathname* pairs.

ARGUMENTS

linkname (required) Specify the link's name and location.

object_name (required) Specify the object to which the link points.

OPTIONS

-p[airwise] Instructs **cr1** to accept the list of tokens as consecutive pairs. This is necessary whether you are using wildcards or not.

-r Replace an existing link. Use this option to change a link's *object_name*.

NOTES

If you use more than one pair of name tokens with this command, you must use the -p option. It instructs the command to accept the list of tokens as consecutive pairs. This is necessary whether you are using wildcards or not. In the past, this command has correctly paired off tokens without the prompting of a switch; now the -p switch must be used to achieve this result. The switch has been added to protect against inadvertent use in a shell, other than /com/sh, where wildcard expansion can be a problem.

EXAMPLES

Create a link called **bugs** in the current working directory.

```
$ cr1 bugs /maintenance/reports
```

Now, when you use **bugs** in a pathname, the command shell substitutes the text /maintenance/reports. Therefore, the pathname

```
bugs/sticky_cursor
```

refers to the same file as the pathname

```
/maintenance/reports/sticky_cursor
```

NAME

crp – create a process on a remote node

SYNOPSIS

crp *-on node_spec [options] [command line]*

DESCRIPTION

crp creates a process on a remote node.

command line (optional)

Specify command line to be executed by the remote process. If the command string contains embedded blanks, enclose it in quotation marks.

Default if omitted: `execute /com/sh`

OPTIONS

The following option, which specifies the remote node, is required:

-on node_spec Specify the remote node on which the process is to be created.

You can specify one of the following options.

-cp (default) Create a remote process running with standard streams connected to the current window. The option is not valid if **-cpo** or **-cps** is specified.

-nwp Do not create a window-pane legend indicating that the local window is connected to a remote process. Use with the **-cp** option only.

-cpo Create a remote process without a connection to the current window, and an identity of `user.none.none`. This option is not valid if **-cp** or **-cps** is specified. To stop these processes, you must first create a visible remote process running the shell, then issue the `sigp` command to stop the background process.

-cps Create a remote process without a connection to the current window, and an identity of `user.server.none`. This option is not valid if **-cp** or **-cpo** is specified. To stop these processes, you must first create a visible remote process running the shell, then issue the `sigp` command to stop the background process.

-n name Specify the name of the remote process. If this option is not specified, the default is `user.id.node_id`. This allows remote processes to be traced to their originator.

-login name [password] Specify the log-in sequence for the remote process on the command line. If the password is omitted, the system prompts you for

it. A null (zero-length) password is specified by the null string ""

Normally `-login` appears with `-cp`. However, you may use `-login` with `-cpo` and `-cps` as well. If `-login` is specified with either `-cpo` or `-cps`, the identity of the created process is the same as that of the caller (as opposed to `user.none.none` or `user.server.none`, respectively). This means that `-cpo` and `-cps` are identical if `-login` is also specified.

If you use `-login` with `-cpo` or `-cps`, you must place both the name and the password on the command line. No prompting is available in this case.

- `-me` Specified instead of `-login`. If `-me` is specified, the created process on the remote node inherits the caller's working directory, naming directory, home directory text string, and SID. This is similar to popping up another shell except that the process is running on another node. If `-me` is specified with either `-cpo` or `-cps`, the identity of the created process is also that of the caller's (as opposed to `user.none.none` or `user.server.none`, respectively). This means that `-cpo` and `-cps` are identical if `-me` is also specified.
- `-quiet` Suppress connection/disconnection messages in the transcript pad.

EXAMPLES

Create a process on node 532 running the shell, and login with the user ID joe.

```
$ crp -on 532 -login joe
```

Create a process on node aef running the shell, and inherit the current process state information.

```
$ crp -on 0aef -me
```

NAME

crpad – create a transcript pad and window

SYNOPSIS

crpad [*options*] [*pathname*]

DESCRIPTION

crpad creates a transcript pad, copies a file (or standard input) into that pad, and then opens a window into the pad. This new pad is not related to the transcript pad attached to processes running the shell; it is for viewing file contents only. This is primarily useful for displaying output being produced inside a pipeline without interrupting the flow of control in the pipe.

You cannot edit transcript pads. If you wish to place a file in a pad for editing, use the **EDIT** key or the **DM** command **ce**.

crpad -input behaves differently. This creates an edit pad and lets you create whatever text you want. When you close the edit pad (with **wc** or the **EXIT** key), that text is copied to standard output.

pathname (optional) Specify the file to be copied into the pad. Not valid if **-input** is used.

Default if omitted: copy standard input

OPTIONS

- i[nput]** Copies data from a temporary edit window to standard output. Not valid if **-tee** or **-pn** are specified.
- p[n] *pathname*** Specify a pathname for the pad. If you specify a pathname, the pad is saved in that file. Note that you can also save the pad after it is created by using the **DM** command **pn** (*pad_name*).
- t[ee]** Copy output to standard output in addition to the new pad.

EXAMPLES

Create a pad that displays the file **test.data**.

```
$ crpad test.data
```

Display the intermediate results in a pipeline.

```
$ $fpat -p '256-' <phone.book | crpad -tee | srf >phone.book.local
```

Create an edit pad. When the pad is closed, sort the text edited and display it in a transcript pad.

```
$ crpad -input | srf | crpad
```

SEE ALSO

More information is available. Type

help pn	For details about saving pads in files
help wc	For details about closing windows
help tee	For details about copying intermediate results in a pipeline into a disk file

NAME

crsubs – create a protected subsystem

SYNOPSIS

crsubs *subsystem_name*

DESCRIPTION

crsubs is used to create a protected subsystem.

A protected subsystem is a set of programs and data files. The programs are called the managers of the protected subsystem; the data objects they manage are said to be owned by the subsystem. Protected subsystems allow you to define exactly how a file can be accessed.

crsubs creates a protected subsystem by creating the subsystem shell and putting it in the directory `/sys/subsys`. Once there, the subsystem shell can be invoked using the **ensubs** command. The access control list for the directory `/sys/subsys` determines who can create new subsystems: whoever has `x` rights to the directory can create new subsystems.

The access control list on the file `/sys/subsys/subsystem_name` determines who can enter the subsystem *subsystem_name*. Whoever has read and execute rights to it can enter the subsystem. This capability should be restricted generally to the creators of the subsystem or to the system administrators.

ARGUMENTS

subsystem_name (required) Specify name of subsystem to be created. This file will contain the subsystem shell and reside in the `/sys/subsys` directory.

EXAMPLES

Create the subsystem **news**ubs.

```
$ crsubs newsubs
```

SEE ALSO

More information is available. Type

help protection For general information about Domain/OS protection mechanisms

help protection protected_subs
For detailed information on protected subsystems

help subs For information on displaying or selecting subsystem attributes

help ensubs For information on entering a protected subsystem

help xsubs For details about executing a shell script as a subsystem manager

help login window For information on the attributes of a log-in window

help protection acls For information on access control lists.

NAME

crty – create a new type

SYNOPSIS

crty [*options*] *type_name*

DESCRIPTION

crty creates a new type. It creates an identifier for the new type, and associates it with the supplied type name. New types are used to identify a new kind of manager for streams.

type_name (required) Specify the name to assign to the created type.

OPTIONS

- n** *node_spec* Specify the node on which the type is to be created. Type **help node_spec** for details about node specification syntax. You may also specify the entry directory of a volume mounted for software installation, as shown in the example below. If this option is omitted, the type is created on the current node.
- l** List the type name/type identifier pair that is created.
- b**[inary] *pathname* Create the type from the specified object module (which was created by **crtyobj**). This allows you to use an object module (shipped on media like floppies, magnetic tapes, etc) to add a new type to a system.
- u** *high.low* Create the type with the specified unique identifier (UID). Give the high and low addresses for the UID as indicated.
 Note: Use this option only for system debugging. Misuse of this option may cause programs to behave incorrectly.

EXAMPLES

```
$ crty example_type -l
"example_type" 24BF9F41.100001FB created.
```

```
$ crty example_type -n //test_vol -l
"example_type" 24BFA6F8.200001FB created on volume //test_vol.
```

In the following example, the disk has been mounted for software installation. The disk's top level directory (cataloged as `/mount_disk` by the `mtvol` command) must contain a `sys` directory. If it does not, you get a "type manager directory not found" error.

```
$ mtvol w /mount_disk
$ /etc/mount /mount_disk
$ crty example_type -n /mount_disk -l
"example_type" 24BFB71E.200001FB
  created on volume //my_node/mount_disk.
```

SEE ALSO

More information is available. Type

help dlty	For information on deleting types
help lty	For information on listing types
help inty	For information on installing new types

NAME

crtyobj – create a type object module for binding

SYNOPSIS

crtyobj [*options*] *type_name* [*variable_name*]

DESCRIPTION

crtyobj creates an object module that contains a global symbol with the type UID. This module is bound with type managers. The variable is passed into calls to **trait_\$mgr_dcl** to declare support for the specified type.

type_name (required) Specify the name of the type for which an object module is to be created.

variable_name (optional)

Specify the variable name for the type UID.

Default if omitted: name the variable *type_name_\$uid*

OPTIONS

-b *bin_name* Specify the output binary file name. The default is *type_name.bin*.

-sect *section_name* Specify the section name for the data area in which the variable is declared. The default section name is *.data*.

-u *high.low* Specify the type UID explicitly with the high and low addresses in the positions indicated.

NOTE: Use this option only for system debugging.

EXAMPLES

```
$ crtyobj example_type example_$uid
```

```
$ bind -b example_mgr example_main.bin example_calls.bin example_type.bin
```

SEE ALSO

More information is available. Type

help crty For information on creating types

help dlty For information on deleting types

help lty For information on listing types

NAME

`csr` – set or display command search rules

SYNOPSIS

`csr` [*directory ...*] [`-a` *dir_name*]

DESCRIPTION

Command search rules determine which directories the shell examines to find commands. `csr` lets you display or change this list. If you run a new shell script, the subordinate shell inherits the search rules of the parent shell. Note that this does not apply to shells you create with the SHELL key, since the SHELL key actually creates a new, separate process. Its shell receives the default search rules described below.

By default, the shell looks for commands in this order:

1. Your working directory ("`.`"), or the directory specified by the command's path-name.
2. Your personal command directory, `~/com` (the `com` subdirectory of your naming directory).
3. The system command directory, `/com`.
4. The directory `/usr/apollo/bin`.

Specifying `csr` without arguments or options displays the current command search rules.

ARGUMENTS

directory (optional) Specify new command search sequence. Multiple directory pathnames are permitted; separate names with blanks. The shell searches the directories in the order that you specify.

Default if omitted: display current search rules unless `-a` is specified

OPTIONS

`-a` *dir_name* Append the specified directory name(s) to the existing command search sequence. This allows you to add a new directory to the end of the list without retyping the entire list. Multiple directory pathnames are permitted; separate names with blanks.

EXAMPLES

Display current search rules.

```
$ csr
~/com /com /usr/apollo/bin
```

Set new search sequence by adding an additional command directory.

```
$ csr . ~/com //us/myproj/com /com
```

Append the directory ~/com/special_commands to the current list of directory names.

```
$ csr -a ~/com/special_commands
```

NAME

ctnode – catalog a node in the network

SYNOPSIS

/etc/ctnode [options] [node_name [net.]node_id ...]

DESCRIPTION

ctnode informs the local node that a remote node exists, thereby enabling network file access to the remote node. The command catalogs the *node_name* in the local copy of the network root directory as the entry directory for the remote node. In other words, **ctnode** adds the directory *//node_name* to your copy of the network root directory.

For information on deleting a *node_name* entry, type **help uctnode**.

We assign a node ID to every node during the manufacturing process. To find out the node ID of a node, type the following command at its keyboard:

```
$ lcnode -me
```

from another node's network root into your own, or any other node's network root. The merge options (**-md** and **-ms**) add the entry for a node to the target, provided the entry does not already exist and the source has exactly one entry for that node. In the case of one source and one target entry that match for a node, those entries are assumed to be correct. All other cases are considered to be ambiguous and the "confusion-resolution protocol" is invoked.

This "confusion-resolution protocol" first attempts to verify the correct entry name with the node itself. If the node is available, the reply from the node is cataloged regardless of whether **-md** or **-ms** is used because an answer from the node itself is assumed to be the truth.

If the node is unavailable to resolve an ambiguity, the entry containing the most recent UID (latest time stamp portion of the UID) is used. In this case, existing entries in the target directory are only updated if the **-ms** option is used. Multiple name/ID pairs are permitted.

If you do not specify **-n**, **-update**, or **-from**, the *node_name* and *net.node_id* arguments are required.

node_name (optional) Specify the name of the node you wish to catalog. If the *net.node_id* argument is specified, then *node_name* is required.

Default if omitted: you must use **-n**, **-update**, or **-from**

[*net.*]*node_id* (optional)

Specify the hexadecimal ID (and optional network ID) of the node you wish to catalog. The node must be connected to the network when this command is executed. If the *node_name* argument is specified, then *node_id* or [*net.*]*node_id* is required.

Default if omitted: you must use **-n**, **-update**, or **-from**

Multiple name/ID pairs are permitted.

OPTIONS

If you do not specify **-n**, **-update**, or **-from**, the *node_name* and [*net.*]*node_id* arguments are required. The **-n**, **-update**, and **merge** options work only for remote nodes running Aegis SR5.0 or later. The [*net.*]*node_id* forms work only when both the local and remote nodes run Aegis SR9.0 or later.

-root Catalog *node_name* as the entry directory name for *node_id* in both the master network root directory and the local copy of the network root directory. This option is valid only if the *node_name* and *node_id* arguments are specified. This option is not valid if the **-n** option is specified.

-n [*net.*]*node_id*... Copy the entry directory name from the network root directory of the specified remote node to the network root directory of the local node. You do not need to know the entry directory name. However, you must specify the *node_id* or the *net.node_id* of the remote node. Multiple *node_id*'s and *net.node_id*'s may be specified. Use this option instead of the *node_name* *net.node_id* argument pair. This option is not valid if the **-r** option is specified.

-update Obtain a list of nodes currently responding to a network inquiry and perform the same operation as **-n** for each node. Names are replaced with the most current version, if they already exist in your local copy of the network root directory, and new names are added.

- from //node ...** Look in the specified list of network root directories for the names to add to the target network root, or use this network root as the source for names to merge into the target network root. Wildcards may be used to specify source node names. The **-from** option is not supported in a Domain internet environment.
- md** This option is used with **-from**. Merges all names in the source network root into the target network root. Preference is given to existing names in the target if there are unresolved conflicts (see the discussion of "confusion-resolution protocol" above).
- ms** This option is the same as **-md**, except that preference is given to entries in the source network root when there are unresolved conflicts (see the discussion of "confusion-resolution protocol" above).
- on //node ...** Catalog names in the network root of the specified nodes instead of the local network root. Wildcards may be used to specify target node names. The **-on** option is not supported in a Domain internet environment.
- r** Replace cataloged names if they already exist. An error occurs if you do not specify this option and try to add a *node_name* that has already been cataloged (unless you are using **-update**).
- l** List node names as they are cataloged.
- idupl** Ignore entry (suppress error messages) if name already exists in the target.
- lc** List invocations and resolutions of the "confusion-resolution protocol".

EXAMPLES

Add the node whose ID is 21 and whose entry directory name is os to your node's catalog:

```
$ /etc/ctnode os 21
```

Bring your node's catalog up to date with any new nodes on the network:

```
$ /etc/ctnode -update
```

Copy names os and eve from the network root on //master.

```
$ /etc/ctnode os eve -from //master
```

Add node ID 21 with the name `os` to the network root of all nodes whose names begin with "a".

```
$ /etc/ctnode os 21 -on //a?*
```

Merge network root of `os` into local network root, resolving conflicts:

```
$ /etc/ctnode -md -from //os
```

NAME

ctob – catalog an object

SYNOPSIS

/etc/ctob pathname uid_hi.uid_low

DESCRIPTION

ctob assigns a pathname to an object that has a known unique identifier (UID). **ctob** catalogs the pathname and associated UID in the naming tree. This command is primarily for system-level debugging.

pathname (required) Specify assigned pathname.

uid_hi.uid_low (required) Specify the high and low portion of the UID as 32-bit hexadecimal numbers.

EXAMPLES

```
$ /etc/ctob lastfile 10A0BAAD.60000102
```

NAME

cv_t_font – convert fonts from pre-SR10 to SR10 format

SYNOPSIS

cv_t_font *destination source1* [*source2*]

DESCRIPTION

The **cv_t_font** command creates a new font file formatted for SR10. If one *source* name is given, it is converted and placed in the *destination* file. If two source names are given, then the characters in the second source font are concatenated with the characters in the first font, converted, then placed in the *destination* font file.

The source font(s) must be in pre-SR10 format. Since all pre-SR10 fonts have space pre-allocated for 128 characters, the new font can contain up to 256 characters.

If the *destination* font file already exists, or if **cv_t_font** fails to find either *source* file, an error is printed, and the command terminates without changing any fonts.

EXAMPLES

The following example takes the **vt100s** font from `/sys/dm/fonts` and formats it for SR10 in the file **vt100s** in the working directory:

```
$ cv_t_font vt100s /sys/dm/fonts/vt100s
```

The following example takes the **courier10** and **courier10.a** font files from `/sys/dm/fonts`, concatenates them, and formats them for SR10 in the file **courier10** in the working directory:

```
$ cv_t_font courier10 /sys/dm/fonts/courier10 /sys/dm/fonts/courier10.a
```

SEE ALSO

More information is available. Type

help tr_font transliterate characters within a font

NAME

`cvt_rec_uasc` – convert file types

SYNOPSIS

`cvt_rec_uasc source_pathname [target_pathname] -ot type [options]`

DESCRIPTION

`cvt_rec_uasc` converts files from type "rec", "hdru", or "uasc" to files of type "rec", "hdru", or "uasc".

Wildcards in pathnames associated with this command are permitted.

ARGUMENTS

source_pathname (required)

Specify the file to be converted.

target_pathname (optional)

Specify file to be created. An error occurs if this file already exists (see `-r` below). The *target_pathname* may be derived. If target is a directory, the source file is converted and placed in that directory.

Default if omitted: the converted file becomes *source_pathname* and the original file renamed *source_pathname.cbak*

`-ot type` (required)

Specify type of file to be created *target_pathname*. Choose one of the following for *type*: "rec", "hdru", or "uasc."

Wildcards in pathnames associated with this command are permitted.

OPTIONS

`-r`

Replace *target_pathname* if it already exists.

EXAMPLES

List current files in specified directory and their types.

```
$ ld -a
```

```
Directory "/larry/cvt_rec_uasc_examples":
```

sys	type	blocks	current				
type	uid	used	length	attr	rights		name
file	rec	1	42	P	pndwrx		a
file	rec	1	42	P	pndwrx		b
file	rec	1	44	P	pndwrx		c

```
3 entries, 3 blocks used.
```

```
$
```

Convert all files to type uasc; suppress wildcard queries.

```
$ cvt_rec_uasc ?* -ot uasc -nq
```

```
$ ld -a
```

```
Directory "/larry/cvt_rec_uasc_examples":
```

sys	type	blocks	current				
type	uid	used	length	attr	rights		name
file	uasc	1	37	P	pndwrx		a
file	rec	1	42	P	pndwrx		a.cbak
file	uasc	1	38	P	pndwrx		b
file	rec	1	42	P	pndwrx		b.cbak
file	uasc	1	40	P	pndwrx		c
file	rec	1	44	P	pndwrx		c.cbak

```
6 entries, 6 blocks used.
```

```
$
```

Convert files named a, b, and c to type rec and write them to a.x, b.x1 and c.x.

```
$ cvt_rec_uasc [a-c] =.x -ot rec -nq  
$ ld -a
```

Directory "/larry/cvt_rec_uasc_examples":

sys	type	blocks	current				
type	uid	used	length	attr	rights		name
file	uasc	1	37	P	pndwrwx		a
file	rec	1	42	P	pndwrwx		a.cbak
file	rec	1	42	P	pndwrwx		a.x
file	uasc	1	38	P	pndwrwx		b
file	rec	1	42	P	pndwrwx		b.cbak
file	rec	1	42	P	pndwrwx		b.x
file	uasc	1	40	P	pndwrwx		c
file	rec	1	44	P	pndwrwx		c.cbak
file	rec	1	44	P	pndwrwx		c.x

9 entries, 9 blocks used.

```
$
```


NAME

cvtname – convert pathnames between upper and lowercase and preserve colons

SYNOPSIS

cvtname [*options*]

DESCRIPTION

Prior to SR10, the colon (:) was used as an escape character for the purpose of storing mixed-case names. For example, the filename "Readme" was stored as ":readme". Domain/OS programs mapped ":r" and interpreted it as "R". In pre-SR10 Aegis-only environments, colons used in pathnames were treated as literal characters, since Aegis was not case sensitive.

Colon-character constructs in pathnames from pre-SR10 file systems are converted to the appropriate uppercase letter (or special character) automatically when they are copied to SR10 systems. The **cvtname** command allows you to selectively undo that process and thereby restore literal colons to pathnames. **cvtname** also allows you to convert pathnames to all uppercase or all lowercase. The tool operates on entire pathnames. That is, you cannot convert one capital letter in an SR10 pathname back to a "colon-character" sequence without converting them all.

Regardless of the mode specified, **cvtname** queries you before converting each pathname, unless you specify **-nq**, in which case the changes are applied to all objects subordinate to the pathname specified.

OPTIONS

Without options, **cvtname** converts capital letters back to colon-character sequences.

- m *pathname*** Convert capital letters in the names of all objects in *pathname* back to colon-character sequences. If **-li** is also specified, potential changes are listed but no changes are made. If **-nq** is present, the changes are done automatically and all modified names are listed. (The default is **-m** without **-nq**)
- l *pathname*** Convert *pathname* and subordinate object names to all lowercase. If **-li** is also specified, potential changes are listed but no changes are made. If **-nq** is present, the changes are done automatically and all modified names are listed.
- u *pathname*** Convert *pathname* and subordinate object names to all uppercase. If **-li** is also specified, potential changes are listed but no changes are made. If **-nq** is present, the changes are done automatically and all modified names are listed.

EXAMPLES

The following example allows you to convert the capital letters or colon-character constructs in pathnames in the directory **leduc**, querying you before making any changes. Output is shown under the command line. The left-hand column shows unconverted name; right shows converted. Type **y** to convert, **n** to keep old name.

```
$ cvtname /ledu
```

```
/ledu/:C /ledu/:::c n
/ledu/CAT /ledu/:c:a:t y
/ledu/CAT converted to /ledu/:c:a:t
```

The following example allows you to selectively convert pathnames in **ledu** to upper-case.

```
$ cvtname --upper /ledu
```

```
/ledu /LEDU n
/ledu/:c /ledu/:C n
/ledu/:c:a:t /ledu/:C:A:T n
/ledu/acl_from_whoiville /ledu/ACL_FROM_WHOVILLE y
/ledu/acl_from_whoiville converted to /ledu/ACL_FROM_WHOVILLE
/ledu/backup.pas /ledu/BACKUP.PAS n
/ledu/ffl /ledu/FFl y
/ledu/ffl converted to /ledu/FFl
/ledu/TD/backup_history /ledu/TD/BACKUP_HISTORY n
```

NAME

cvtrgy – convert registry between SR9.x and SR10 formats

SYNOPSIS

```
cvtrgy [-from9to10 | -from10to9 [ -favor_etc ] ] -readonly |  
-owner pgo | -first | -nq | -from source_rgy -to dest_rgy
```

DESCRIPTION

The **cvtrgy** command allows the system administrator to generate an SR10 format registry database from SR9.7 registry files, or generates SR9.7 registry files with data from the SR10 registry. The tool operates on SR9.7 nodes only. Both the **rgyd** and **llbd** servers must be running on the SR10 node, except when the **-first** option is used. Run **cvtrgy** the first time when you add SR10 nodes to your network, and periodically thereafter to keep the pre-SR10 and SR10 registry information synchronized.

You must specify either **-from9to10** or **-from10to9**. By default, **cvtrgy** creates a read-only registry of the destination type. That is, **cvtrgy -from9to10** creates a read-only SR10 format master registry, while **cvtrgy -from10to9** creates a read-only SR9.x format master registry. You can then propagate the information to replica registries in the appropriate way.

Whenever the conversion from SR10 to SR9 occurs, if the registry files exist at the destination node specified in the command line, the tool quits without updating. This means that before running **cvtrgy -from10to9**, you should rename (or move) the SR9.x registry database on the destination node.

The **cvtrgy** tool assigns UNIX identifiers automatically during the conversion process if you prefer. However, if your pre-SR10 node runs Domain/OS, you should preserve the identifiers associated with accounts in your current (pre-SR10) **/etc/passwd** and **/etc/group** files. In normal operation, **cvtrgy** looks for the **/etc/passwd** and **/etc/group** files and assigns identifiers from them, if they exist. Therefore, you should run **cvtrgy** on a 9.7 node that either contains your master **/etc/passwd** and **/etc/group** files or has a link to them.

If **cvtrgy** doesn't find the **/etc/passwd** and **/etc/group** files and an **/etc** directory exists, it queries you before assigning new UNIX identifiers, unless the **-nq** (no query) flag is turned on, in which case **cvtrgy** exits with an error.

In order to add or change accounts and other registry data, you must edit the writable registry with the tool appropriate to the registry's format (i.e., with **edrgy** on SR10, **edacct** and **edppo** on SR9.x) on a node running the same software release as the format of the writable registry. Thus, if your SR9.x registries were writable, you'd have to edit them using **edacct** and **edppo**, from a node running SR9.7. Once your SR10 registry is the writable one, use **edrgy**.

The **cvtrgy** tool resides in the **/install/tools/cvtrgy** after an SR10 installation and must be copied to an SR9.7 node before you run it. After running **cvtrgy**, you must also run the **crpasswd** command on an SR9.x node to update the **/etc/passwd** and **/etc/group** files. The SR10 directory **/install/tools** contains a new version of **crpasswd** which you

should copy to all SR9.7 nodes that need to run `crpasswd`. (You can rename or replace the old version of `crpasswd`.) See the SR10 Transition Guide for further details on running `cvtrgy`.

OPTIONS

- `-from9to10` Convert SR9.x registry files to SR10 registry format
- `-from10to9` Convert SR10 registry data to SR9.7 format and place in SR9.7 registry files
- `-from source_rgy` Specify source for registry data to be converted. For `-from9to10`, must be in the form `//node_name/registry/rgy_site`. For `-from10to9`, must be `//node_name`. Either or both registry sites may be remote from the node running `cvtrgy`.
- `-to dest_rgy` Specify destination for converted registry data. For `-from9to10`, must be in the form `//node_name/registry/rgy_site`. For `-from10to9`, must be `//node_name`. Either or both registry sites may be remote from the node running `cvtrgy`.
- `-owner pgo` Specify SR10 registry owner, in the SID form `p.g.o`, where all `pgo` names and the `pgo` account already exist in the SR9.7 registry. `pgo` is a string of the form `pers.group.org`. You must specify with every invocation of `-from9to10`. This option is meaningful only with the `-from9to10` option.
- `-first` Specify that this is the first invocation of `cvtrgy`. In this case only, `cvtrgy` runs without `rgyd` and `llbd` servers running. Use only once. Only meaningful with `-from9to10`.
- `-readonly` Make SR9.7 registries read-only, permanently. Only meaningful with `-from9to10`. Can only be run in this mode once; after running, cannot use `-from9to10` again.
- `-nq` No query. Silent mode. Don't query before assigning new UNIX identifiers (`cvtrgy` quits). Don't query for owner (`cvtrgy` quits).
- `-favor_etc` If you've edited UNIX IDs (numbers) in the SR9.7 `/etc/passwd` or `/etc/group` after you've already run `cvtrgy` at least once, you should propagate the new numbers to the SR10 registry. Running `cvtrgy` with this option, in the `-from9to10` direction, propagates the new UNIX IDs to the SR10 registry. After running `cvtrgy` with this option, you must also run `/etc/syncids` on all SR10 disks. Only meaningful with `-from9to10`.

CONVERTING FROM SR9.7 TO SR10

You must be root to run `cvtrgy`. Use the following command line. The `node_name1` is the SR9.7 node.

```
$ cvtrgy -from9to10 -from //node_name1/registry/rgy_site
      -to //node_name2 -owner pgo -first
```

CONVERTING FROM SR10 TO SR9.7

The person who runs the tool must be logged in as root or locksmith. Use the following command line. The `node_name1` is the SR10 node.

```
$ cvtrgy -from10to9 -from //node_name1 -to //node_name2/registry/rgy_site
```

EXAMPLE

The following is a sample transcript from a `cvtrgy` session that converts SR9.x registry data files to an SR10 format registry database. This is the first time `cvtrgy` has been run on the network. A single collision is shown to illustrate `cvtrgy`'s warning message format; you may see more warnings at your site.

```
$ cvtrgy -from9to10 -from //dog/registry/rgy_site1 -to //cat -first -owner %sys_admin.%
```

Phase 1 - opening registry files:

Phase 2 - modifying SR9 registry files:

```
Converted person file saved in registry
//dog/registry/rgy_site1
```

```
Converted project file saved in registry
//dog/registry/rgy_site1
```

```
Converted org file saved in registry
//dog/registry/rgy_site1
```

Phase 3 - converting person file:

```
?(cvtrgy) Warning - unix id collision:
person bin_sr9 reassigned from 3 to 10002
Converted person file saved in registry
//dog/registry/rgy_site1
```

Phase 4 - converting project file:

?(cvtrgy) Warning - unix id collision:
project backup reassigned from 1001 to 3
Converted project file saved in registry
//dog/registry/rgy_sitel

Phase 5 - converting org file:

Converted org file saved in registry
//dog/registry/rgy_sitel

Phase 6 - converting accounts:

Phase 7 - adding default accounts:

Converted account file saved in registry
//dog/registry/rgy_sitel

Phase 8 - closing the sr9 registry files:

Phase 9 - writing conversions to sr10 registry:

Conversion completed successfully:

NAME

date – display the current date and time

SYNOPSIS

date [*options*]

DESCRIPTION

date prints the current system date and time. It requires no arguments or options. If no options are specified, the date is displayed as shown in Example 1 below. Dates in European languages are displayed using the 24-hour clock.

The hardware date and time may be set with the calendar command.

OPTIONS

-y	Display year as yyyy.
-md	Display month and day as mm/dd.
-t	Display time in 24-hour format (hh:mm:ss).
-d	Display year, month, and day.
-dmy	Display date as dd/mm/yyyy.
-f[rench]	Display date and time information in French.
-g[erman]	Display date and time information in German.
-i[talian]	Display date and time information in Italian.
-dan[ish]	Display date and time information in Danish.
-n[orwegian]	Display date and time information in Norwegian.
-s[wedish]	Display date and time information in Swedish.
-sp[anish]	Display date and time information in Spanish.
-fi[nnish]	Display date and time information in Finnish.

DATE

Aegis

DATE

EXAMPLES

\$ date

Tuesday, May 3, 1988 4:20:15 pm (EDT)

\$ date -t

15:36:14

\$ date -d

1988/05/03

\$ date -f

mardi, 5 janvier 1988 14:49:11

NAME

dcalc – evaluate logical and arithmetic expressions

SYNOPSIS

dcalc [-h] [*pathname*...]

DESCRIPTION

dcalc mimics the features of a desk calculator, evaluating both logical and arithmetic expressions.

ARGUMENTS

pathname (optional) Specify input file containing expressions to be evaluated, one expression per line.

Default if omitted: read standard input; stop with CTRL/Z

OPTIONS

If no options are specified, all operations are decimal-based.

-h Specify hexadecimal operations.

Expressions

Input expressions can be simple arithmetic expressions or variable assignment expressions. **dcalc** writes the value of each evaluated expression on standard output. Variables hold temporary values, which **dcalc** does not automatically write.

Expressions may include any of the operators listed below in order of precedence:

1. + - Unary plus and negation operators. These may appear only at the start of an expression or within parentheses.
2. << >> Logical left and right shift
3. ** Exponentiation
4. * / % Multiply, divide, modulo (remainder)
5. + - Add, subtract
6. == Equal to
- != Not equal to
- > Greater than
- >= Greater than or equal to
- < Less than
- <= Less than or equal to

- | | |
|------|-------------------|
| 7. ! | Unary logical not |
| 8. | Logical or |
| & | Logical and |
| ^ | Logical xor |

Relational operators return the value 1 for true and 0 for false. `dcalc` performs operations in double precision floating point, except for logical operators listed as items 2 and 8 above, which use 32-bit integers.

Variables

Expressions may include previously declared variables. Use this format to declare a variable: `name = expression`

- A variable name must begin with a letter and may consist of any combination of letters and digits.
- `dcalc` does not automatically print replacement expressions, because they usually contain temporary values.

Radix Control

You can change the default base for input or output using `ibase` (input base) and `obase` (output base) statements. For example,

```
ibase = 2
```

```
obase = 16
```

causes `dcalc` to interpret input in binary and print results in hexadecimal.

To set an individual number's radix, precede it with the desired radix and a pound sign. For example,

```
16#100
```

specifies the hexadecimal number 100 (equals 256 in decimal).

EXAMPLES

Your input: dcalc output:

10 + (-64 / 24)** 6

temp = 2#101
temp == 5 1 (true)

ibase = 16
obase = 2
11 + 28 111001
1a + 0f 101001

Note that when you type a hexadecimal number that begins with a letter, you must precede it with a zero.

ibase = 16
numa = 100
numb = 100
numa + numb 512

NAME

dde – Domain Distributed Debugging Environment

SYNOPSIS

```
dde [-do "cmd_list"]
      [ [-on target_machine] [-target_type target_type]
        { [-input pathname] [-output pathname [-ao]]
          [-errors pathname [-ae]] program_invocation
        | -attach process_id } ]
```

DESCRIPTION

The **dde** command invokes the Domain Distributed Debugging Environment, the standard debugger for the Domain/OS operating system at SR10. For complete information about this debugger and its commands, consult the *Domain Distributed Debugging Environment Reference* (011024) or invoke the debugger's own **help** command for online assistance.

OPTIONS

- do "*cmd_list*"** Execute *cmd_list* (a list of debugger commands) before executing any startup files or debugging the program. The sample option specification **-do "property layout -notarget"** illustrates a common use of this option (to inhibit the creation of a separate window for the target program).
- on *target_machine*** Debug the program or process on the specified target machine, where *target_machine* is a node name or node ID.
- target_type *target_type***
Specify the type of target machine; *target_type* must be "m68k" for SR10.
- input *pathname*** Read target program input from *pathname*.
- output *pathname* [-ao]**
Direct target program output to *pathname*. With **-ao**, append output to *pathname*.
- errors *pathname* [-ae]**
Direct target program error output to *pathname*. With **-ae**, append error output to *pathname*. To redirect error output and standard output to the same file, use the same *pathname* on both options or use "&1" as an argument to the **-errors** option.
- program_invocation*** Invoke *program_invocation* (the *pathname* of an executable image, plus any arguments) for debugging. This specification must be last on the **dde** command line.

-attach *process_id* Attach to a running process identified by the UNIX pid *process_id*. Use the `/bin/ps` or `/com/pst -un` commands to get the pid of a process.

NAME

dldupl – strip repeated lines from a file

SYNOPSIS

dldupl [-c] [*pathname* ...]

DESCRIPTION

dldupl reads the input file(s), comparing adjacent lines. Second and succeeding copies of repeated lines are removed; the remaining lines are written to standard output.

ARGUMENTS

pathname (optional)

Specify input file. Multiple filenames permitted; separate names with blanks.

Default if omitted: read standard input

OPTIONS

-c Write number of occurrences of each line to standard output.

EXAMPLES

Suppose you have two alphabetized dictionary files. To create one dictionary file containing the words from both, use:

```
$ srf -m dict1 dict2 | dldupl >dict.new
```

This merges the words from the two files (*srf -m*), then deletes any duplicate words and saves the result in the new dictionary.

NAME

dlf – delete one or more files

SYNOPSIS

dlf [*pathname...*] [*options*]

DESCRIPTION

dlf deletes the file(s) specified. To delete objects other than files, see **dll** (**delete_link**) and **dlt** (**delete_tree**).

ARGUMENTS

pathname (optional) Specify file to be deleted. Multiple names and wildcarding are permitted; separate names with blanks.

Default if omitted: read names from standard input

OPTIONS

- f** Force file deletion if you have owner rights, even if you don't have delete rights.
- l** List names of deleted files.
- du** Delete when unlocked. If the object to be deleted is locked when **dlf** is invoked, the delete operation is performed when the object is unlocked.

EXAMPLES

```
$ dlf mary.bak -l
(file) "mary.bak" deleted.
```

SEE ALSO

More information is available. Type

- help dll** For details about deleting links
- help dlt** For details about deleting directory trees

NAME

dll – delete a link

SYNOPSIS

dll *pathname* ... [*options*]

DESCRIPTION

dll deletes a link. After execution of this command, the link is no longer available for use.

ARGUMENTS

pathname (required) Specify *pathname* of the link to be deleted. Multiple pathnames and wildcarding are permitted; separate names with blanks.

OPTIONS

-l List name(s) of link(s) as deleted.

EXAMPLE

Delete the link **bugs** from the current working directory.

```
$ dll bugs
```

```
$
```

SEE ALSO

More information is available. Type

help dlf For details about deleting files

help dlt For details about deleting directory trees

NAME

dlt – delete a tree

SYNOPSIS

dlt *pathname* ... [*options*]

DESCRIPTION

dlt deletes the directory named by the *pathname*, and all its descendants in the naming tree.

ARGUMENTS

pathname (required) Specify directory to be deleted. If *pathname* is a directory, **dlt** deletes the directory and all subordinate objects (subdirectories, files, and links). If a link, **dlt** deletes the link name, but has no effect on the files and directories named by the link. Multiple *pathnames* and wildcarding are permitted.

OPTIONS

-l List files, links, and directories as they are deleted.
-ld List directories as they are deleted.
-lf List files as they are deleted.
-ll List links as they are deleted.
-f Force object deletion if you have owner rights, even if you don't have delete rights.
-du Delete when unlocked. If the object to be deleted is locked when **dlt** is invoked, the delete operation is performed when the object is unlocked.
-pr *pathname* Preserve specified *pathnames*.

You can combine **-ld**, **-lf**, and **-ll** to create the type of listing you desire.

EXAMPLES

Delete the two directory trees specified.

```
$ dlt april_backup may_backup
```

DLT

Aegis

DLT

SEE ALSO

More information is available. Type

help dlf For details about deleting only files

help dll For details about deleting only links

NAME

dlty – delete a type

SYNOPSIS

dlty [*options*] *type_name*

DESCRIPTION

dlty deletes a type and any installed type manager.

type_name (required) Specify the name of the type to be deleted.

OPTIONS

-n *node_spec* Specify the node on which the type is to be deleted. Type **help node_spec** for details about node specification syntax. You may also specify the entry directory of a volume mounted for software updates, as shown in the example below. If you omit the **-n** *node-spec* the type is deleted on the current node.

-l List the type name/type identifier pair that is deleted.

EXAMPLES

```
$ dlty example_type -l
"example_type" 24BF9F41.100001FB deleted.
```

```
$ dlty example_type -n //test_vol -l
"example_type" 24BFA6F8.200001FB
deleted from volume //test_vol.
```

In the following example, the disk has been mounted for software updates. The disk's top level directory (cataloged as **/mount_disk** by the **mtvol** command) must contain a "sys" directory. If it does not, you get a "types file not found" error.

```
$ mtvol w /mount_disk
$ dlty example_type -n /mount_disk -l
"example_type" 24BFB71E.200001FB deleted
from volume //my_node/mount_disk.
```

SEE ALSO

More information is available. Type

help crt For information on creating types

help lty For information on listing types

NAME

dlvar – deletes all of the specified variables

SYNOPSIS

dlvar *var_name* ...

DESCRIPTION

The **dlvar** command deletes the variable(s) specified. If a variable had another value at a higher level of invocation, the variable is restored to that value.

ARGUMENTS

var_name ... (required) Specify the variable name to be deleted. Multiple names are permitted, separated by blanks.

NAME

dmtvol – dismount a logical volume

SYNOPSIS

dmtvol *ddevice*[*unit*] [*log_vol_number*] [*pathname*] [*options*]

DESCRIPTION

dmtvol dismounts a logical volume that was previously mounted with the **mtvol** (**mount_volume**) command. After the volume has been dismounted, it is unavailable for further access.

ARGUMENTS

ddevice (required) Specify the type of disk on which the volume resides: *w* for a Winchester disk, *s* for a storage module, or *f* for a floppy disk.

unit (optional) Specify a unit number (0 or 1 only) for the device, if necessary. For example, *s1* denotes storage module unit 1.

Default if omitted: 0 (zero)

log_vol_number (optional)

Specify the number of the logical volume to be dismounted.

Default if omitted: 1

pathname (optional) Specify the entry directory of the logical volume. If you include this argument, **dmtvol** dismounts the volume and uncatalogs its entry directory. If you omit it, **dmtvol** dismounts the logical volume, but retains its name in the naming tree.

OPTIONS

-fu Forcibly unlock any locked objects, then dismount the volume. If you omit this option, the dismount fails if the volume contains any locked objects.

-nw Prevents **dmtvol** from trying to write to the disk during the dismount. Normally, writing to the disk saves current information. However, if the disk was removed prior to the dismount, you should use this option.

EXAMPLES

Dismount storage module unit zero, logical volume 2, and leave its name in the naming tree.

```
§ dmtvol s 2
```

Dismount floppy unit zero, logical volume 1, and delete its name from the naming tree.

```
$ dmtvol f/floppy
```

SEE ALSO

More information is available. Type

help mtvol For details about mounting logical volumes

NAME

drm_admin – Data Replication Manager Administrative Tool

SYNOPSIS

/etc/ncs/drm_admin

DESCRIPTION

The **drm_admin** tool administers servers based on the Data Replication Manager (DRM), such as the Global Location Broker (GLB).

It can inspect or modify replica lists, merge databases to force convergence among replicas, stop servers, and delete replicas.

The role of **drm_admin** is to administer the replication of databases, not to change the data they contain. For instance, you can use **drm_admin** to merge two replicas of the GLB database, but you must use **lb_admin** to add a new entry to the database. Also, although **drm_admin** can stop or delete a GLB replica, you must invoke **glbd** (the GLB daemon) directly if you want to start or create a replica.

Once invoked, **drm_admin** enters an interactive mode, in which it accepts the commands described in the following section.

COMMANDS

Most **drm_admin** commands operate on a default object (*default_obj*) at a default host (*default_host*). Together, *default_obj* and *default_host* specify a default replica. Defaults are established by the set command and are remembered until changed by another set.

Currently, the only known object is **glb**.

Some **drm_admin** commands operate on a host other than the default; we identify this host as *other_host*. The host name you supply as a *default_host* or an *other_host* takes the form *family:host*. The only currently supported *family* is **dds**; you can specify a host in this family by its entry directory or by its network address. For example, **dds://thurber** and **dds:#1234.abcd** are acceptable host names.

addrep *other_host* Add *other_host* to the replica list at *default_host*. The replica at *default_host* will propagate *other_host* to all other replica lists for *default_obj*.

delrep *other_host* [-force]
Delete the replica of *default_obj* at *other_host*.

The **delrep** command tells the replica at *other_host*

1. To propagate all of the entries in its propagation queue
2. To propagate a delete request to all other replicas, causing *other_host* to be deleted from all other replica lists for *default_obj*

3. To delete its copy of *default_obj*

4. To stop running

The `-force` option causes a more drastic delete. It deletes *other_host* from the replica list at *default_host*. The replica at *default_host* propagates the delete request to the replicas at the hosts remaining on its list, thereby removing *other_host* from all other replica lists for *default_obj*.

A force delete can cause data to be lost and should only be used when a replica has irrevocably "died." We recommend strongly that you do a `merge_all` operation after the force delete to prevent the remaining replicas of the *default_obj* database from becoming inconsistent. If the deleted replica is still running, it should be reset.

info Get status information about the replica for *default_obj* at *default_host*.

lrep [-d] [-clocks] [-na]
List replicas for *default_obj* as stored in the replica list at *default_host*.

The `-d` option lists deleted as well as existing replicas.

The `-clocks` option shows the current time on each host and indicates clock skew among the replicas.

The `-na` option lists the network address of each host.

merge { -from | -to } *other_host*

The `merge` command copies entries in the *default_obj* database and replica list from one replica to another. It copies an entry if no corresponding entry exists in the destination database or if the corresponding entry in the destination database bears an earlier time stamp.

A merge does not cause entries to be propagated. The database and replica list at the origination are not changed.

The `-from` option copies entries from the *default_obj* database and replica list at *other_host* to the *default_obj* database and replica list at *default_host*.

The `-to` option copies entries from the database and replica list at *default_host* to the database and replica list at *other_host*.

A merge `-from` followed by a merge `-to` causes the replicas at the two hosts to converge.

- merge_all** The `merge_all` command uses *default_host* as the hub for a global merge of all replicas for *default_obj*. A `merge_all` first does a `merge -from` each host on *default_host*'s replica list; then it does a `merge -to` each host on the replica list. All replicas of *default_obj* are thereby forced into a consistent state.
- A `merge_all` should be used
- When a replica is force deleted
 - When a replica is reset
 - When a replica has been incommunicado for 2 weeks or more
 - When a replica "dies" (for example, when its database is destroyed by a disk failure). The `merge_all` operation does not cause any entries to be propagated.
- monitor [-r n]** This command causes `drm_admin` to read the clock of each replica of the *default_obj* every *n* minutes and to report any clock skews or non-answering replicas. If you do not specify `-r`, the period is 15 minutes.
- quit** Quit the `drm_admin` session.
- reprep other_host** Replace the network address for *other_host* in the replica list at *default_host*. The replica at *default_host* will propagate the new entry for *other_host* to all other replica lists for *default_obj*. Use `reprep` only when a host's network number changes.
- reset other_host** Reset the replica of *default_obj* at *other_host*.
- The `reset` command tells the replica at *other_host* to delete its copy of *default_obj* and to stop running. It does not cause *other_host* to be deleted from any other replica lists. This command can cause data to be lost unless a successful `merge_all` is done first.
- set [-o obj_name] -h host_name** Set the default object and host. Subsequent commands that do not specify a host will be sent to this host. All subsequent commands will operate on the object *obj_name*. If you do not specify the `-o` option, `drm_admin` keeps the current *default_obj*.
- If you use `set` with the `-o` option, `drm_admin` checks the clocks at all hosts with replicas of the specified object.
- stop** Stop the server for *default_obj* that is running at *default_host*.

EXAMPLES

Start `drm_admin`, set the default object to `glb`, and set the default host to `//mars`:

```
$ /etc/ncs/drm_admin
drm_admin: set -o glb -h dds://mars
      Default object: glb  default host: dds://mars
                        state: in service
      Checking clocks of glb replicas
      dds://mars      1987/04/09.17:09
      dds://pluto    1987/04/09.17:09
      dds://mercury 1987/04/09.17:07
```

SEE ALSO

`glbd`, `lb_admin`

Managing the NCS Location Broker.

NAME

dspst – display process status graphically

SYNOPSIS

```
dspst [-r n] [-p] [-L1] [-os] [-m]
        [-io] [-a] [-n node_spec]
        [-large|-small]
```

DESCRIPTION

dspst displays process statistics in a graphical, bar-chart fashion within the current process window. The chart is updated periodically (see **-r** below). The default action of this command is to display the brief Domain/OS process list, all user processes and all I/O information in a font size automatically selected based on window size.

While **dspst** is running, the following keys are interpreted as follows:

All Keyboards:

CRTL/T	Move to top
CRTL/B	Move to bottom
RETURN	Exit
CRTL/N	Exit
CRTL/Y	Exit and save current image
Boxed up arrow	Scroll backward 1/2 window
Boxed down arrow	Scroll forward 1/2 window
Shifted up arrow	Scroll backward 1 line
Shifted down arrow	Scroll forward 1 line
EXIT or ABORT	Exit
SAVE	Exit and save current image

OPTIONS

-r <i>n</i>	Specify that the display should be repeatedly updated every <i>n</i> seconds. If this option is omitted, the display is updated every 4 seconds.
-p	Show process information.
-ll	Show Domain/OS and user-process information.
-os (default)	Show brief Domain/OS and full user-process information.
-m	Show missing CPU time.
-io (default)	Show I/O statistics.
-a	Show all information (same as -ll -io -m).
-n <i>node_spec</i>	Specify remote node whose process statistics are to be listed.

- large (default)** Force use of large font for display.
-small Force use of small font for display.

EXAMPLES

1. Display Domain/OS, user process, and I/O status.

```
$ dspst
```

2. Display Domain/OS, user process, and I/O status for the node named //fred using the large font.

```
$ dspst -n //fred -large
```

SEE ALSO

More information is available. Type

- help pst** For information on displaying process status in a non-graphic format.

NAME

dtcb – dump contents of tcp control blocks

SYNOPSIS

```
/etc/dtcb [-f] [[-t]
            [<tcbl_addr>|-a ] |-u
            [<ucbl_addr>|-a ]]
```

DESCRIPTION

The command **dtcb** dumps the contents of the tcp control blocks associated with a particular tcp connection. The address of the tcb to be dumped may be obtained using the **netstat** program. Two control blocks are dumped: the ucb (user control block) which contains the send and receive queues and user-related flags, and the tcb (tcp control block) which contains the connection sequence numbers, state, flags, and out-of-sequence queues.

OPTIONS

```
-f                Force output if tcpd not running.
-t <tcbl_addr>   Hexadecimal address of a tcb or, if not supplied, all tcbs.
-u <ucbl_addr>   Hexadecimal address of a ucb or, if not supplied, all ucbs.
-a                All (both tcb's and ucb's for each socket).
```

EXAMPLES

The dump of a tcp control block for a listening ftp connection might look like this:

```
$ /etc/dtcb -t 1A9A50
ucb at 0x1A99C4:
local 0.0.0.0 lport 21
host 0.0.0.0 fport 0
uc_snd 8192 uc_ssize 0 uc_rcv 8192 uc_rsize 0
uc_shead 0 uc_stail 0 uc_rhead 0 uc_rtail 0
xflag:
UREUSEADDR UCANACCEPT
iostate:

status:
UCLOSED
flags:
UTCP
oobmark 0 oobcnt 0
```

```
TCB at 0x1A9A50:
lport 0x0  fport 0x0
t_state LISTEN
irs 00000000 rcv_urp 00000000 rcv_urg 00000000 rcv_nxt 00000000
    rcv_end 00000000
iss 1E3202D4 seq_fin 1E3202D4 snd_end 1E3202D4
    snd_urp 00000000 snd_lst 00000000

snd_nxt 1E3202D4 snd_una 1E3202D4 snd_wl 00000000
    snd_hi 1E3202D4

rex_val 00000000 rtl_val 00000000 xmt_val 00000000
flags:
snd_wnd 0  maxseg 0  xmtime 2  rxtct 0
timers:
INIT 0  REXMT 0  REXMTTL 0  PERSIST 0  FINACK 0
t_rcv_next 1A9A50  t_rcv_prev 1A9A50
```

NAME

ed – invoke line editor

SYNOPSIS

ed [-n] [*pathname*]

DESCRIPTION

ed invokes the line editor. Input text and editing commands are read from standard input. While you may use ed to create text files interactively, it is better suited for use in programs and scripts. Use the EDIT key or the DM command, ce, to create and edit files interactively.

ARGUMENTS

pathname (optional) Specify the file to be edited. ed reads the file into a buffer for editing and remembers its name for future use. ed operates on the buffer copy; changes made there have no effect on the original file until you issue a w (write) command from within ed. Files must be less than 6400 lines and less than 256,000 characters.

If you omit the *pathname* argument, the edit buffer is empty and no filename is remembered for future use. You must specify an explicit filename when you exit the editor.

Default if omitted: see above

OPTIONS

-n Suppress the printing of line counts by the e (edit), r (read), and w (write) commands.

SUMMARY OF ED COMMANDS

Addresses:

17	A decimal number.
.	The current line.
\$	The last line of the file.
/pat/	Search forward for a line containing <i>pat</i> .
\pat\	Search backward for a line containing <i>pat</i> .
line+n	<i>n</i> lines forward from <i>line</i> .
line-n	<i>n</i> lines backward from <i>line</i> .

Defaults:

(.)	Use the current line.
(.+1)	Use the next line.
(.,.)	Use the current line for both line numbers.
(1,\$)	Use all lines.

Commands:

(.) A	Append text after line (text follows).
(.,n Bn	Browse over the next <i>n</i> lines (default <i>n</i> is 22). If <i>n</i> is negative, print the last <i>n</i> lines before the current line. If <i>B</i> is specified, print <i>n</i> lines with the current line in the center of screen.
(.,.) C	Change text (text follows).
(.,.) D	Delete text.
E file	Discard the current text, enter <i>file</i> , remember filename.
F	Print filename.
F file	Remember filename.
(.) I	Insert text before the line (text follows).
(.,) Kline	Copy text to a new line after the specified line.
(.,) Mline	Move text to a line after the specified line.
(.,) P	Print text. (You can append this to other commands.)
Q	Quit.
(.) R [file]	Read <i>file</i> , appending after the current line.

(.,) <i>S</i> / <i>pat/new/GP</i>	Substitute <i>new</i> for leftmost <i>pat</i> . (<i>G</i> implies all occurrences.)
(1,\$) <i>W</i> [<i>file</i>]	Write the <i>file</i> ; leave the current text unaltered. (If you do not specify a file, write to current filename.)
(.) =[<i>P</i>]	Print the line number and current line.
(.+1) < <i>CR</i> >	Print the next line.
(1,\$) <i>G</i> / <i>pat/command</i>	Execute <i>command</i> on lines containing <i>pat</i> (except <i>A</i> , <i>C</i> , <i>I</i> , and <i>Q</i> commands).
(1,\$) <i>X</i> / <i>pat/command</i>	Execute <i>command</i> on lines not containing <i>pat</i> (except <i>A</i> , <i>C</i> , <i>I</i> , and <i>Q</i> commands).
# ...	Comment.
<i>\$n</i>	Read or write temporary buffer, <i>n</i> .

ed prints the error message "?" whenever it does not understand or fails to execute a command

NOTE

There is a homonymous DM command: ed -- delete the character preceding the cursor.

LIMITATIONS

- Files being edited can contain up to 6400 lines.
- When a global search and substitute combination fails, the entire global search stops.
- Problems sometimes occur when you use @*n* to remove or insert newline characters, especially in global commands.

SEE ALSO

More information is available. Type

help ed commands	For detailed information about each ed command
help patterns	For information about the pattern-matching scheme
help ce	For information on creating and editing files interactively
help ed_dm	For information on the synonymous DM command

NAME

edacl – edit or list an ACL

SYNOPSIS

edacl [*commands*] [*options*] *pathname*...

DESCRIPTION

Every directory and file has an associated access control list (ACL) that lists users and their rights to the object. **edacl** edits or displays the ACL of the object(s) specified. The structure and usage of an ACL is described in detail in **help protection acls**.

ARGUMENTS

pathname (required) Specify the object whose ACL you wish to edit or display. Multiple pathnames and wildcarding are permitted.

commands (optional) Specify the action(s) described below. If you do not specify a command, **edacl** enters an interactive editing mode.

Default if omitted: read commands from standard input; do not precede commands with a hyphen (-) in this mode.

COMMANDS

Many of the commands described below take arguments called 'sid' and 'rights'. These are summarized in the sections preceding the **EXAMPLES**.

- l** List ACL entries.
- a *sid rights*** Add the specified entry to an ACL. You receive an error message if the ACL entry exists.
- af *sid rights*** Add force. Add the specified entry to an ACL. You do not receive an error message if the ACL entry exists.
- ar *sid rights*** Add the specified rights to an ACL. You receive an error message if the entry does not exist.
- c *sid rights*** Change the access rights in the entry for *sid* (replaces current rights). You receive an error message if the entry does not exist.
- cf *sid rights*** Change force. Change the access rights to an ACL. You do not receive an error message if the entry does not exist.
- d *sid*** Delete the ACL entry for *sid*. You receive an error message if the entry does not exist.
- df *sid rights*** Delete force. Delete the specified rights from the entry for *sid*. You do not receive an error message if the ACL entry does not exist.
- dr *sid rights*** Delete the specified rights from the entry for *sid*. You receive an error message if the entry does not exist.

- p *p rights*** Set the required entry for person *p*.
- g *g rights*** Set the required entry for group *g*.
- o *o rights*** Set the required entry for organization *o*.
- w *rights*** Set the required entry for world.
- lao** Restrict access to local node.
- nolao** Remove restriction to local node.
- recalc** Recalculate statrights for an ACL. This command is provided to allow the Aegis user to undo the effects of **chmod**. These rights are recalculated automatically any time that **edacl** changes the ACL for an object.
- q** Quit without changing the object's ACL. This command is useful only when you supply **edacl** commands interactively (see **-inter**). To signal successful completion and update the ACL, use EOF in standard input (usually CTRL/Z).

The following three commands are meaningful primarily for Domain/OS applications. If the pertinent index is enabled, the process executing the file assumes the person, group, and/or organization identity of the file. Each may be set only to the corresponding required entry. For example, you may only setuid to the owner of the file. (This is the Domain/OS equivalent of Aegis protected subsystems.) The indexes may be set for both files and directories, but are meaningful only for files.

- setuid [off | on]** Assign the set person index.

If you specify **off**, the set person index is deleted.
If you specify **on**, the set person index is added.
- setgid [off | on]** Assign the set group index.

If you specify **off**, the set group index is deleted.
If you specify **on**, the set group index is added.
- setoid [off | on]** Assign the set organization index.

If you specify **off**, the set organization index is deleted.
If you specify **on**, the set organization index is added.

OPTIONS

- dir** Operate only on directories.
- file** Operate only on files.
- id** Edit the default initial ACL for directories (**-dir** implied).
- if** Edit the default initial ACL for files (**-dir** implied).

The following two options apply only when `edacl` reads commands from standard input:

- prog** `edacl` interprets commands when it receives an EOF (usually CTRL/Z). This is the default when you redirect standard input (i.e., instructed the program to read commands from a shell program, here document, file, or pipe).
- inter** `edacl` interprets commands as you enter them. This is the default when you have not redirected standard input. You may only specify one pathname (with no wildcards) in this mode. `edacl` changes a copy of the ACL; the command does not assign a new ACL to an object until it reads an EOF. Thus, `edacl -inter` does not change an ACL if you terminate the session with the "q" command.

Description of SIDs

An SID (subject identifier) is the mechanism used to identify users to the system when they log in. Basically, an SID has three parts: a person name (p), group name (g), and organization name (o); the combination is often abbreviated to 'pgo'.

SIDs consist of the p, g and o separated by periods. Thus

```
joe.sftwr.r_d
```

might be the name of a software programmer in the R & D organization. His person name is 'joe'; his group name is 'sftwr'; his organization name is 'r_d'.

In ACLs, SIDs may contain one or more wildcards, similar in concept to wildcards used with pathnames. A '%' in the person, group, or organization part of an SID will match any person, group or organization (respectively). Thus

```
joe.%.%
```

matches user 'joe' regardless of his group or organization names.

Description of Rights

A complete description of the various protection rights is available in

\$ help protection rights

The following are the basic kinds of operations that can be performed on objects, and the rights which allow them when present in an ACL entry.

For all objects:

- p** Protect rights; allows rights to be changed.

For files:

- w** Write rights; allows file to be written.
- r** Read rights; allows file to be read.
- x** Execute rights; allows file to be executed.
- k** Keep rights; prevents an object from being deleted or from having its name changed.

For directories:

- w** Write rights; allows names to be added, changed or deleted.
- r** Read rights; allows directory to be listed.
- s** Search rights; allows directory to be searched for subordinate objects.
- x** Execute rights (synonym for search rights).
- k** Keep rights; prevents an object from being deleted or from having its name changed.

For initial file/initial directory ACLs:

- i** Inherit rights. The SID portion of a required entry is inherited from the creating process. This would normally only be used if someone needs to inherit the SID portion and does not wish to inherit rights from the current process (see `-inh_all`).

The following abbreviations exist for sets of rights:

-owner	Gives all rights. For files, it means: <code>prwx</code> For directories: <code>prwx</code>
-user	Gives all rights except ability to change ACL. For files, it means: <code>wrx</code> For directories: <code>wrx</code>
-read	For files, allows reading; can't change ACL. Precisely, it means: <code>r</code>
-exec	For files, allows reading, execution; can't change ACL. Precisely, it means: <code>rx</code>
-ldir	For directories, allows listing; can't change ACL. Precisely, it means: <code>rx</code>
-adir	For directories, allows adding names and links, and listing; can't change ACL. Precisely, it means: <code>wrx</code>
-none	Gives no rights, for files or directories. Used to explicitly deny rights to specific SIDs that would otherwise be granted rights because they are members of a group or organization. Delete and rename rights come from directories. This means that if you set <code>-none</code> rights on a file, but do not set the same rights for the directory that contains the file, your file is NOT protected from being deleted. You must set <code>k</code> (keep) rights to protect a file in a non-protected directory.
-ignore	For required entries: is used to specify that the required entry for an object is not to be used in rights checking.
-inh_rights	For directory initial ACLs: specifies rights are to be inherited from the current process.
-inh_all	For directory initial ACLs: specifies both rights and pgo information is to be inherited from the current process.

EXAMPLES

The order of the commands in the following sequence is significant.

```

$ edacl -l sales                                List ACL for the file 'sales'.

Required entries
none.%.%           [ignored]           No person listed
%.none.%          [ignored]           No group listed
%.%.none          [ignored]           No organization listed
%.%.%            prwx-                Others have prwx access to file
Extended entry
rights mask:      -----

$
$ edacl sales -o r_d -rx -l                    Give r_d read and execute access.

Required entries
none.%.%           [ignored]           No person listed
%.none.%          [ignored]           No group listed
%.%.r_d           -r-x-                r_d has read and execute access to file
%.%.%            prwx-                Others have prwx access to file
Extended entry
rights mask:      -----

$
$ edacl sales -p mary -owner -l                Indicate an owner.

Required entries
mary.%.%          prwx-                Owner
%.none.%          [ignored]           No group listed
%.%.r_d           -r-x-                r_d has read and execute access to file
%.%.%            prwx-                Others have prwx access to file
Extended entry
rights mask:      -----

```

```

$ edacl sales -w -none -l

```

Deny access to all others (e.g. world).
Note that the directory must also be set to `-none`, otherwise the file is not protected from deletion or renaming.

```

Required entries
mary.%.%      prwx-
%.none.%     [ignored]
%.%.r_d      -r-x-
%.%.%        -----
Extended entry
rights mask:  -----
$
$ edacl sales -a jill -owner -l

```

Add jill to the ACL for sales with all rights

```

Required entries
mary.%.%      prwx-
%.none.%     [ignored]
%.%.r_d      -r-x-
%.%.%        -----
extended entries listed below
Extended entry
rights mask:  prwx-
Extended entries
jill.%.%     prwx-
$
$ edacl sales -p joe -owner -l

```

Make user joe be the owner instead of mary

```

Required entries
joe.%.%      prwx-
%.none.%     [ignored]
%.%.r_d      -r-x-
%.%.%        -----
Extended entry
rights mask:  prwx-
Extended entries
jill.%.%     prwx-
$
$ edacl sales

```

Interactive mode.

```

*g osdev wrx

```

Allow users in the osdev group to change file contents, but do not

let them assign rights to others (no p).

```
*l
Required entries
joe.%.%      prwx-
%.osdev.%    -rwx-
%.%.r_d      -r-x-
%.%.%        -----
Extended entry
rights mask: prwx-
Extended entries
jill.%.%     prwx-
```

Additional rights

```
$
$ edacl sales -w r
$ edacl -l sales
Required entries
joe.%.%      prwx-
%.osdev.%    -rwx-
%.%.r_d      -rwx-
%.%.%        -r---
Extended entry
rights mask: prwx-
Extended entries
jill.%.%     prwx-
$
```

Change everyone else's access to read only. Note that the more liberal rights (wrx) assigned to osdev, joe and r_d still apply, since specific entries override general ones.

```
$ edacl sales -c jill wrx
$ edacl -l sales
Required entries
joe.%.%      prwx-
%.osdev.%    prwx-
%.%.r_d      -rwx-
%.%.%        -r---
Extended entry
rights mask: -rwx-
Extended entries
jill.%.%     -rwx-
$
```

Change jill's rights to remove right to change ACL.

SEE ALSO

More information is available. Type:

help protection acls For a detailed description of ACLs.

help acls For a list of commands used to manipulate ACLs.

help protection For a general discussion of Domain/OS protection mechanisms.

help protection sids For details about subject identifiers (pgo's).

help protection rights

For details about the various access rights and what they mean.

NAME

edfont – edit a character font

SYNOPSIS

edfont [*file* | *-v*]

DESCRIPTION

edfont is an interactive program with both menu-driven and command-line interfaces. It allows you to create, edit, and view character font files. You can specify the font file with the *file* parameter, or use the “Open Font” entry in the “File” menu. If the *-v* option is used, **edfont** will print its version number and exit.

Generally, you must press the left mouse button <M1> to activate commands in the menu-driven interface. When you must enter a string (for example, when you designate which font you want to open) and there is a “Done” field on the menu, enter the string, point to “Done” and press <M1> to activate. If “Done” does not appear when you must enter a string, simply type the string and press <RETURN> to activate the command.

When using the menu-driven interface, you may notice that you cannot always select every menu choice. For example, you can't select “Open Font” if you already have one open, and likewise it's invalid to try to close a font when no font is open. When commands are invalid, as in these cases, their places on the menus are grayed out so that they can't be selected.

edfont lets you select a character (glyph) in a variety of ways. The utility interprets input this way:

- Any three-character string whose first character is a lowercase *c* has its final two characters interpreted as a compose sequence (e.g., *ca`* for lowercase *a* with a circumflex accent)
- Any string that begins with *0x* is interpreted as a hexadecimal code (e.g., *0x41* for uppercase *A*)
- Any string that begins with *0* (zero) is interpreted as octal (e.g., *0101* for *A*)
- Any string that begins with a digit other than zero is considered to be decimal (e.g., *65* for *A*)
- Any other string is considered to be an ASCII character (e.g., *A* for *A*)

For more information on compose sequences, see your system's User's Guide. For a list of decimal, octal, and hexadecimal values for the characters in Apollo's default character set, as well as a list of the compose sequences, see the files in the */usr/pub* directory.

When you invoke **edfont**, it sets default values for several variables. You can change those defaults using either the appropriate command in the menu-driven interface or set in the command-driven interface. For more information on these interfaces see the section on command interfaces, below.

The following table lists variables, their types, default values (if any), and purpose.

Variable/Type	Default	Description
fontpath/string	:/sys/dm/fonts	List of directories, separated by colons, in which edfont should search for fonts
fontservers/string	/usr/apollo/lib/edfont	The search path for the font servers directory
fill/string	outline	The name of the current fill pattern
fontorigin/coord	none	The coordinate value that tells the number of pixels below and to the left of the font origin
fontsize/coord	none	The width and height of the font bounding box
fontspacing/coord	none	The horizontal and vertical font spacing (leading)
glyphoffset/coord	none	The offset of the current glyph from the font origin
glyphsize/coord	none	The width and height of the bitmap for the current glyph
glyphwidth/coord	none	The number of pixels from the right edge of the current glyph to the left edge of the next glyph
mask/string	src ^ dst	The current mask (raster operation)

edfont handles fonts created using Apollo's current and pre-SR10 formats, as well as Adobe BDF fonts.

Menu Interface

Note: You can get additional information about any item on the display by pressing the HELP key at the cursor position where you need help. This pops a help box. To return to the original display, move the cursor out of the help box.

Font When you position the cursor here and press <M1>, edfont displays a menu with the following choices:

- Open Font
- Close Font
- Select Glyph
- Font Params
- Glyph Params
- Quit

Use these choices to open and close the font you want to edit, select an individual glyph (character) to edit, and examine or change the font's parameters or a single glyph's parameters.

Tools If you press <M1>, you will see the following choices:

Grid
Metrics

By default, both are turned on. If you turn off Grid, you no longer will see the pixel-by-pixel bitmap grid in the edit window. If you turn off Metrics, the glyph fills the edit window.

Metrics shows these three attributes of your glyph and font:

- Origin and baseline (fine dotted line)
- Glyph-bounding box (long dashed line)
- Font-bounding box (short dashed line)

Commands If you press <M1>, you will see the following choices:

Undo	Undo remembers your last 10 changes to the current glyph. Undo does not work on parameter changes, however.
Run Commands	You can set up a file of commands and direct edfont to execute that file. For more information on the commands you can use, see the description of the command interface, below.
Copy Glyph	Copies a glyph from elsewhere in your font or from another font.
Delete Glyph	This deletes a glyph.
Rotate Glyph	This rotates a glyph by the number of degrees you specify.
Draw	When you position the cursor here and press <M1>, you will see the following choices:
Pixel	Manipulate individual pixels
Freehand	Draw freehand
Line	Draw lines
Box	Draw boxes
Circle	Draw circles
Cut	Select and delete a pixel or range of pixels
Copy	Select and copy a pixel or range of pixels

Paste Paste in a pixel or range of pixels that you have previously cut or copied

Zoom Zoom in on a selected portion of the glyph

Note that after you Cut or Copy, edfont automatically changes the Draw mode to Paste. You can manually change it to something else if you prefer.

Fill When you position the cursor here and press <M1>, you will see the following choices:

Outline this is the default
 25% gray
 50% gray
 75% gray
 black
 bricks
 chex
 /stripes right-leaning stripes
 \stripes left-leaning stripes
 lstripes vertical stripes
 -stripes horizontal stripes
 tri
 waves

The way edfont fills an entity such as a circle or box depends on which fill you choose. If you choose 50% gray, for example, and then create a box, edfont turns on half of the pixels inside the box to create a 50% gray effect. If you choose 75% or 25% gray, edfont turns on proportionally more or fewer pixels to get the desired effect.

Mask When you position the cursor here and press <M1>, you will see the following choices (where "src" means source, "dst" means destination, and the other characters are logical operators):

Menu Choices	Logical Operation
clear	Assign zero to all new destination values
src & dst	Assign source AND destination to new destination
src & ~dst	Assign source AND complement of destination to new destination
src	Assign source values to new destination
~src & dst	Assign complement of source AND destination to new destination

<code>dst</code>	Assign all destination values to new destination
<code>src ^ dst</code>	Assign source EXCLUSIVE OR destination to new destination (default)
<code>src dst</code>	Assign source OR destination to new destination
<code>~(src dst)</code>	Assign complement of source AND complement of destination to new destination
<code>src == dst</code>	Assign source EQUIVALENCE destination to new destination
<code>~dst</code>	Assign complement of destination to new destination
<code>src ~dst</code>	Assign source OR complement of destination to new destination
<code>~src</code>	Assign complement of source to new destination
<code>~src dst</code>	Assign complement of source OR destination to new destination
<code>~(src & dst)</code>	Assign complement of source OR complement of destination to new destination
<code>set</code>	Assign 1 to all new destination values

Setting the mask value turns pixels on. That is, if you select a pixel or range of pixels with this mask, all the pixels turn black, regardless of whether they already were black. The mask clear turns a pixel or range of pixels off (white), regardless of the pixel's initial value.

The default mask `src ^ dst` toggles pixels. That is, if they already were black, they become white, and vice versa. However, if you are drawing in Freehand mode, this mask toggles the first pixel you cross and then sets the rest of the pixels you cross to that first pixel's value.

When you have a font open, the menu-driven interface also includes two boxes on the right side of the display labeled “<<<” and “>>>”. The two are for displaying the previous and next glyph, respectively, in the current font. Move the cursor over either box and press <M1> to activate.

Command Interface

In addition to edfont's menu-driven interface, you can use the following commands in the input pad at the bottom of the edfont window, or embed them in edfont scripts.

Commands (Arguments)	Description
<code>!shell-command</code>	Run a shell command in the edfont window.
<code>box x1 y1 x2 y2</code>	Draw a box that is bounded by (x1,y1) and (x2,y2).
<code>circle x y r</code>	Draw a circle which has its center at (x,y) and a radius of r.

<code>close [-save -nosave]</code>	Close the font. If you specify <code>-save</code> , edfont saves your changes, while if you specify <code>-nosave</code> , edfont ignores them.
<code>copy glyphcode [fontfile]</code>	Copy the specified glyph to the current glyph. If you specify a <i>fontfile</i> , edfont copies the glyph from that font; otherwise, it copies the glyph from the current font.
<code>delete</code>	Delete the current glyph.
<code>grid on off</code>	Turn the bitmap grid on or off.
<code>help [command]</code>	Get a list of available commands, or get help on the specified command.
<code>line x1 y1 x2 y2</code>	Draw a line that begins at $(x1,y1)$ and ends at $(x2,y2)$.
<code>metrics on off</code>	Turn the font metrics display on or off.
<code>next</code>	Go to the next glyph in the current font.
<code>open fontfile</code>	Open the specified fontfile.
<code>pixel x y</code>	Draw a pixel at (x,y) .
<code>previous</code>	Go to the previous glyph in the current font.
<code>quit [-save -nosave]</code>	Exit edfont, closing the current font (if one is open). See <code>close</code> for information on <code>-save</code> and <code>-nosave</code> .
Commands (Arguments)	Description
<code>rotate degrees</code>	Rotate the current glyph by the specified number of degrees.
<code>select glyphcode</code>	Go to the specified glyph. For information on entering a glyph or <i>glyphcode</i> see the Description section above.
<code>set var=value</code>	Set <i>var</i> to the specified value. <i>var</i> can be one of the edfont's parameters, as described in the Description section above.
<code>source filename</code>	Execute the command-script <i>filename</i> .
<code>undo</code>	Undo the last bitmap operation.
<code>unzoom</code>	Zoom out one level.
<code>zoom x1 y1 x2 y2</code>	Zoom in so that the view is filled with the box bounded by $(x1, y1)$ and $(x2,y2)$.

SEE ALSO

More information is available. Type

help fl For information on loading a font

help fonts For information on fonts supplied with the Domain/OS system

NAME

edmtdesc – edit magtape descriptor file

SYNOPSIS

edmtdesc {*options*} *pathname*

DESCRIPTION

edmtdesc allows you to create, list, and modify the magnetic tape descriptor object. The descriptor file provides information to the streams manager so that it can handle subsequent tape operations.

pathname (required) Specify name of magtape descriptor file to be created, listed, or edited.

OPTIONS

At least one of the following options must be specified.

- c** Create a new magtape descriptor object with the name given in the *pathname* argument.
- l [*var...*]** List the values of the variable(s) specified. If no variables are named, the entire magtape descriptor is listed.
- s [*var value*]...** Set the variable(s) indicated to the specified value(s). At least one variable/value pair is required if **-s** is specified. Multiple variable/value pairs are permitted, separated by blanks.

Variables

The variables known to **edmtdesc** are listed below, along with their types and default values. The variable types are: integer (int), Boolean (y/n), character string of *n* letters (c [*n*]), and date (in format yy/mm/dd.hh:mm).

Name	Type	Default	Definition
dev	c[1]	m	Device type ('m' for magtape, 'c' for cartridge)
u	int	0	Magtape unit number (normally 0)
lab	y/n	yes	'Yes' if magtape is ANSI labeled, 'no' if unlabeled
reo	y/n	no	'Yes' to reopen previously used volume, 'no' to open new volume ('yes' suppresses rewind)
clv	y/n	yes	'Yes' closes volume when file is closed, 'no' leaves volume open

Name	Type	Default	Definition
spos	y/n	no	'Yes' saves volume position when volume is closed (for reopen), 'no' rewinds volume when closed
vid	c[6]	-auto	Volume identifier (labeled volumes)
vacc	c[1]	-auto	Volume accessibility (labeled volumes)
own	c[14]	-auto	Volume owner (labeled volumes)
f	int*	1	file sequence number: integer or "cur" for current file, or "end" for new file at end of labeled volume
rf	c[1]	D	record format -- "f" for fixed length, "d" for variable length, "s" for spanned, "u" for undefined
bl	int	2048	block length, in bytes
rl	int	2048	(maximum) record length, in bytes
ascl	y/n	yes	'Yes' for ASCII newline handling (strip newlines on write, supply them on read), 'no' for no newline handling
fsect	int	1	File section number (labeled volumes)
fid	c[17]		File identifier (labeled volumes)
fsid	c[6]		File set identifier (labeled volumes)
gen	int	1	Generation of file (labeled volumes)
genv	int	1	Generation version of file (labeled volumes)
cdate	date	-auto	Creation date of file (labeled volumes)
edate	date	-auto	Expiration date of file (labeled volumes)
facc	c[1]		File accessibility (labeled volumes)
sysc	c[xx]		System code (labeled volumes)
sysu	c[xx]		System use (labeled volumes)
boff	int	0	Buffer offset (labeled volumes, should be 0)

For cartridge tape (dev c), you must change the block length (bl) and the record length (rl) to be 512 or less and the record format to be fixed ("rf f").

EXAMPLES

Edit file set_tape; set the tape unit number to 1; declare tape as ANSI labeled.

```
$ edmtdesc set_tape -s u 1 lab yes
```

Create descriptor file ct for cartridge tape, blocking 4 records of maximum length 128 to each block.

```
$ edmtdesc ct -c -s dev c bl 512 rl 128 rf f
```

SEE ALSO

More information is available. Type

help magtape For general information on magnetic tape usage

help cartridge For general information on cartridge tape usage and support

NAME

edns – invoke editor for **ns_helper**

SYNOPSIS

/etc/edns *[[net.]node_id]*

DESCRIPTION

edns allows you to inspect and/or modify **ns_helper**'s master network root directory and replica list. Once invoked, **edns** enters an interactive mode and accepts the commands described in **help edns** commands.

[net.]node_id (optional) Set the default **ns_helper** to the **ns_helper** at the node specified by the internet address.

Default if omitted: Set the default **ns_helper** to any active **ns_helper**. An **ns_helper** becomes active after its database has been initialized.

SEE ALSO

More information is available. Type

help edns commands

For a summary of **edns** commands

NAME

edrgy – edit the network registry database

SYNOPSIS

`/etc/edrgy [-a | -p | -g | -o] [-l] [-s //site] [-synch] [-v]`

DESCRIPTION

The **edrgy** tool views and edits information in the registry database. You can invoke **edrgy** from any node.

Though anyone can read information in the registry database, you can usually change information only if you own the affected database entries. For example, only the owner of a group can add a name to the group's membership list.

With **edrgy**, you can edit and view names, accounts, and policies in the network registry, as well as entries in the local registry. The tool operates in one of four domains: person names, group names, organization names, and accounts.

OPTIONS

You can specify only one of **-a**, **-p**, **-g**, and **-o**.

- a** (default) Edit or view accounts.
- p** Edit or view persons.
- g** Edit or view groups.
- o** Edit or view organizations.
- l** Edit or view entries in local registry.
- s** Use the specified registry site.
- synch** Synchronize local registry with network registry.
- v** View selected entries.

Unless you specify the **-v** option, **edrgy** operates interactively. The following sections describes the commands you can enter in the interactive mode.

COMMANDS FOR PERSONS, GROUPS, AND ORGANIZATIONS

`v[iew] [name | number] [-f] [-m] [-po]`

View *name* entries.

If you specify a *number*, **edrgy** displays all matching entries, including any aliases.

The **-f** option displays entries in full (all fields except the membership list and organization policy).

If you are viewing groups or organizations, **-m** displays the membership list. For persons, **-m** lists all groups of which the person is a member, including groups that cannot appear in a project list.

If you specify `-po` while viewing organizations, `edrgy` displays policy information. Otherwise, it shows only the name and the UNIX number.

```
a[dd] [ person number [ fullname ] [ -al ] [ -o owner ] ]
a[dd] [ group number [ fullname [ password ] ] [ -nl ] [ -o owner ] ]
a[dd] [ organization number [ fullname [ password ] ] [ -o owner ] ]
```

Create a new name entry.

If you do not specify a *person*, *group*, or *organization* name, the `add` command enters an interactive mode and prompts you for each field in the entry. If you are adding organizations in the interactive mode, the command prompts you for policy information as well.

Specify the *owner* as a *person.group.organization* triplet. You can use `%` as a wildcard for any or all of the components. If you do not use the `-o` option, `edrgy` assigns the default owner, which you can set or display with the `defaults` command.

For persons, the `-al` option creates an alias entry. If *number* (the UNIX number) is already assigned to a person and you do not specify `-al`, an error occurs and you must either choose a different *number* or specify `-al`. If you use `-al` to create an alias and *number* is not already associated with a primary name, `edrgy` issues a warning but creates the alias.

For groups, the `-nl` flag indicates that the group is not to be included on project lists; omitting this flag allows the group to appear on project lists.

For groups and organizations, a space between quotation marks indicates a nil password.

Use quotation marks to embed spaces (or quotation marks) in a *fullname*. A single space between quotation marks indicates a nil *fullname*.

```
c[hange] [ person [ -n name ] [ -u number ] [ -f fullname ] [ -o owner ]
          [ -al | -pr ] ]
c[hange] [ group [ -n name ] [ -u number ] [ -f fullname ] [ -o owner ]
          [ -p password ] [ -nl | -l ] ]
c[hange] [ organization [ -n name ] [ -u number ] [ -f fullname ] [ -o owner ]
          [ -p password ] ]
```

Change a *name* entry.

If you do not specify a *person*, *group*, or *organization* name, the `change` command enters an interactive mode and prompts you for a name. If

you do not specify any fields, the command prompts you for each field in succession. To leave a field unchanged, press <RETURN> at the prompt. If you are changing organization entries in the interactive mode, the command prompts you for policy information as well.

For person entries, the `-al` flag changes a primary name into an alias, while the `-pr` flag changes an alias into a primary name. This change can be made only from the command line, not in the interactive mode.

For group entries, the `-nl` flag disallows the group from appearing in project lists, while the `-l` flag allows the group to appear in project lists.

For organization entries, you can change policy information only in the interactive mode.

A single space between quotation marks indicates a nil *fullname* or *password*.

Specify the *owner* as a *person.group.organization* triplet. You can use `%` as a wildcard for any or all of the components.

Changes to a person name are reflected in membership lists that contain the person name. For example, if the person *ludwig* is a member of the group *composers* and the person name is changed to *louis*, the membership list for *composers* is automatically changed to include *louis* but not *ludwig*.

Changes to *number* (the UNIX number) cause the operating system to change its mapping of the UID, the primary name, and any aliases from the old *number* to the new one. However, files owned by the old *number* do not automatically show the new *number* as their owner.

The only fields of reserved entries that you can change are the *fullname*, the *password*, the *owner*, and (for *groups*) the property that allows a *group* to appear in project lists. If a reserved *group* is allowed to appear in project lists, you can disallow it; but if the *group* is disallowed, you cannot allow it.

```
m[ember] [ group | organization [ -a member_list ] [ -r member_list ] ]
```

Edit the membership list for a group or organization.

If you do not specify a group or organization, the `member` command enters an interactive mode and prompts you for names to add or remove.

The `-a` flag precedes the person names (separated by spaces) to be added to the membership list, while the `-r` flag precedes those to be removed. If you do not include either flag on the command line, `edrgy` prompts you for names to add or remove.

Adding a person to a membership list permits creation of a login account for that person with that group or organization.

Removing *person* from the membership list for *group* has the side effect of deleting all login accounts of the form *person.group*, and likewise for organizations.

```
del[ete] { person | group | organization }
```

Delete a name entry.

You cannot delete reserved names. Deleting a group or organization has the side effect of deleting any accounts with that group or organization.

```
adopt uid_high.uid_low person number [ fullname ] [ -o owner ]
adopt uid_high.uid_low group number [ password [ fullname ] ] [ -nl ] [ -o owner ]
adopt uid_high.uid_low organization number [ password [ fullname ] ] [ -o owner ]
```

Create a primary name entry for the specified UID.

The UID must be an orphan (a UID for which no name exists in any domain). The *uid_high* and *uid_low* are hexadecimal numbers.

An error occurs if you specify a name or UNIX number that is already defined within the same domain of the database.

A single space between quotation marks indicates a nil *fullname* or *password*.

Specify the *owner* as a *person.group.organization* triplet. You can use `%` as a wildcard for any or all of the components. If you do not use the `-o` option, `edrgy` assigns the default *owner*, which you can set or display with the `defaults` command.

COMMANDS FOR ACCOUNTS

In all of the account operations, the *account* argument is a *person.group.organization* triplet such as *jones.graphics.research*. Unless otherwise specified, any or all of the components can be the wildcard character, `%`. For example, `view %.dev.%` views all accounts associated with the group *dev*.

In an *account* argument, if you omit a trailing *organization* (or *group.organization*), `%` (or `%.%`) is assumed. Thus, `keats.%.%`, `keats.%`, and `keats` are equivalent.

v[iew] [*account*] [**-f**]

Display login accounts specified by the *account* pgo (*person, group, organization*) triplet.

Without the **-f** flag, view displays only the user fields in each account entry: abbreviated account S encrypted password, miscellaneous information, home directory, and login shell.

With **-f**, view displays the full entry, including the administrative fields as well as the user fields. Administrative information includes who created the account, when it was created, who last changed it, when it was last changed, when it expires, whether it is valid, whether the password is valid, and when the password was last changed.

a[dd] [*account* [**-a** { **p** | **pg** | **pgo** }] [*password* [*misc* [*homedir* [*shell*]]]] [**-pnv**] [**-x** *account_exp* | **none**] [**-anv**]]

Create a login account.

Specify *account* as a pgo triplet. Wildcards are not allowed. If you do not supply an *account* on the command line, **add** enters an interactive mode and prompts you for each field in succession.

If the person specified in *account* is not already a member of the specified group and/or organization, **edrgy** automatically attempts to add the person to the membership lists. If you are not an owner of the group and/or organization, the attempt will fail and the account will not be created.

The **-a** flag indicates the degree of abbreviation allowed for login: **p** means that only the person is required; **pg** means the person and the group; **pgo** means that all three components of the account SID are required. (Of course, a user can always supply more components than are required.) If the abbreviation you specify is already defined for another account, **edrgy** automatically uses the shortest unique abbreviation and issues a warning.

For example, if you create an account **babar.elephants.none** with the abbreviation **p**, a user need only enter **babar** at the login prompt to use the account. If you then create an account **babar.kings.none**, the **p** abbreviation will conflict with the existing account, so the **pg** abbreviation, **babar.kings**, will be the shortest unique one.

Omitting the **-a** is equivalent to specifying **-a p** and results in use of the shortest unique abbreviation.

The *password* must adhere to the policy of the associated organization or the policy of the registry as a whole, whichever is more restrictive.

The *misc* field is not used by the operating system. The *gecos* field of each account's entry in the */etc/passwd* file is the concatenation of the person's full name and the account's *misc*. Use quotes to include spaces, hyphens, or quotes in *misc*.

The *homedir* and *shell* are pathnames. The default *homedir* is */*. The default *shell* is the null string.

Use a single space between quotation marks to indicate a nil *password*, *misc_info*, *homedir*, or *shell*.

The *-pnv* (password not valid) flag specifies that at the next login (for a newly created account, the first login), the user must change the password. If you omit this option, the password is valid.

The *-x* flag sets an expiration date for the account; the default is none.

The *-anv* (account not valid) flag specifies that the account is not currently valid for login. If you omit this option, the account is valid.

```
c[change] [ account [-n new_account] [-a { p | pg | pgo } ]
          [-p password] [-m misc] [-h homedir] [-s shell] ]
          [-pnv | -pv] [-x account_exp | none] [-anv | -av ]
```

Change one or more account entries.

Specify *account* as a pgo triplet. Wildcards are allowed, unless you use the *-n* option. If you do not supply an *account* on the command line, *change* enters an interactive mode and prompts you for each field in succession. Press <RETURN> to leave a field unchanged.

The command line arguments are largely the same as those of the *add* command. The *-n* flag enables you to change the account SID to *new_account*, a pgo triplet that cannot contain wildcards. The *-pv* flag specifies that the password is valid. The *-av* flag specifies that the account is valid.

You can enter a single space between quotation marks to indicate a nil *password*, *misc*, *homedir* or *shell*.

del[ete] *account*

Delete the entry for *account*, a pgo triplet that cannot contain wildcards.

MISCELLANEOUS COMMANDS

do[main] [p | g | o | a]

Change or display the type of registry information being viewed or edited.

You can specify **p** for persons, **g** for groups, **o** for organizations, or **a** for accounts. If you supply no argument, **edrgy** displays the current domain.

s[ite] [//site] [-l]

Change or display the registry site being viewed or edited.

If you specify a *//site*, **edrgy** attempts to use the registry server at the named site. If you specify **-l**, **edrgy** uses the local registry. If you supply no argument, **edrgy** displays the current site.

prop[erties]

Change and/or display the registry properties and policies.

This command prompts you for any changes to make. Press <RETURN> to leave information unchanged.

synch[ronize]

Update the local registry to match the master registry.

If a matching entry cannot be retrieved from the network registry, the local entry is marked invalid for login, and its UNIX numbers are updated.

co[py] [*account*]

Copy information for the specified accounts from the master registry to the local registry.

The *account* is a pgo triplet that can contain wildcards; trailing wildcard components can be omitted. If a matching account already exists in the local registry, **edrgy** updates the information to match that in the master registry; otherwise, **edrgy** adds the entry. If all entries in the local registry are used, **copy** reports an error and terminates.

def[aults]

Change and/or display the default values that **edrgy** uses.

h[elp] [*command*]

Display usage information for edrgy.

If you do not specify a particular command, edrgy lists the available commands.

q[uit] Exit edrgy.

COMMANDS VALID FOR THE LOCAL REGISTRY

To edit or view the local registry, use the **-l** flag when you invoke edrgy. This section lists the commands that are valid for editing or viewing the local registry. Unless otherwise specified, all options are as described in the previous command descriptions.

v[iew] [*name* | *number*] [**-f**] [**-po**]

View name entries. (The **-m** option is not valid.)

v[iew] [*account*] [**-f**]

Display specified login accounts.

c[hange] [*account* [**-a** { *p* | *pg* | *pgo* }] [**-m** *misc*] [**-h** *homedir*] [**-anv**]

Change one or more account entries. (The **-p**, **-s**, **-pnv**, **-pv**, **-x**, and **-av** options are not valid.)

del[ete] *account*

Delete an account entry.

do[main] [*p* | *g* | *o* | *a*]

Change or display the type of registry information being viewed or edited.

s[ite] [*//site*] [**-l**]

Change or display the registry site being viewed or edited.

prop[erties]

Change and/or display the registry properties and policies.

synch[ronize]

Update the local registry to match the master registry.

co[py] [*account*]

Copy information for the specified accounts from the master registry to the local registry.

def[aults]

Change and/or display the default values that edrgy uses.

h[elp] [*command*]

Display usage information for edrgy.

q[uit] Exit edrgy.

NAME

edsd – edit mail subscriber directory

SYNOPSIS

edsd [*options*]

DESCRIPTION

edsd is used to create or modify electronic mail accounts in the subscriber directory. The subscriber directory is used to associate a mail address with a user account name in the network registry. Valid person, group, and organization names must have been previously defined with **edrgy**.

While all the **edsd** options are described below, it is unlikely that you can manipulate the subscriber directory unless you are the network administrator for your network. The subscriber directory is protected by ACL restrictions. However, you can list registry entries.

See *DPSS/Mail User's Guide*, for information about setting up communication between DPSS/Mail) and UNIX delivery subsystems.

OPTIONS

At least one of the following options must be specified.

- l** [*name*]... List subscriber directory entries for the specified *names*. If no *names* are specified, all subscriber directory entries are listed.
- lf** [*name*]... List the *name(s)* specified, along with associated full name text, if any. If *name* is omitted, all names are listed.
- a** *name address* Add a new mail subscriber with the address as specified. The mail address must be a string without embedded blanks or commas. The mail address is as expected by the mail delivery subsystem in use. The name specified must not already be a mail subscriber.
- c** *name address* Change the mail address of an existing mail subscriber to the address as specified. The mail address must be a string without embedded blanks or commas. The mail address is as expected by the mail delivery subsystem in use. The name specified must already be a mail subscriber.
- d** *name* Delete the mail subscriber from the subscriber directory. The name is no longer be considered a mail subscriber by cooperating mail delivery subsystems.

EXAMPLES

List all entries in the subscriber directory.

```
$ edsd -l
      Name                Address
      max                 max@unix
      sam                 sam@dpss
      dan                 dan@mktg.alis
```

Add a new mail subscriber

```
$ edsd -a eli eli@unix
Add:  name="eli"          address="eli@unix" Added.
```

Change mail address

```
$ edsd -c eli eli@dpss
Change: name="chase"     address="eli@dpss" Deleted.
```

NAME

edstr – edit a stream

SYNOPSIS

edstr [-n] { *command* | **-e** *command* | **-f** *cmdfile* ...} [*pathname*]

DESCRIPTION

edstr copies the named input files to standard output, performing editing as directed by **edstr** commands in the command line or in the named command file.

ARGUMENTS

If neither the **-e** nor the **-f** argument is specified, **edstr** assumes that the first token on the command line without a hyphen is an **edstr** command (see below) and that the remaining tokens (if any) are pathnames.

command (optional) Specify a single **edstr** command (except **a**, **c**, **i**, or **r**). **edstr** accepts the **ed** commands **a**, **c**, **d**, **i**, **p**, **r**, **s**, **w**, and **=**. To use the **a**, **c**, **i**, or **r** commands, place them in a command file as described below.

Default if omitted: use **-e** and/or **-f**

The following two arguments may be repeated and intermixed in any order. **edstr** executes them in the order they appear on the command line.

-e *command* (optional)

Specify an **edstr** command (except **a**, **c**, **i**, or **r**). To use the **a**, **c**, **i**, or **r** commands, place them in a command file as described below. **edstr** can accommodate commands totaling approximately 5000 characters (including text arguments), and lines up to 120 characters long.

Default if omitted: use *command* or **-f**

-f *cmdfile* (optional) Specify a file containing **edstr** commands, one per line. Control is passed to this file for command processing. See **-e** for **edstr** command restrictions.

Default if omitted: use *command* or **-e**

pathname (optional) Specify input file to be edited. Multiple pathnames are permitted.

Default if omitted: edit standard input

OPTIONS

- n** Suppress writing of output except for **p** and **w** edstr commands. By default, edstr writes each line of input to standard output after editing. If the **-n** option is specified, it must precede any arguments on the command line.

COMMANDS

Addresses:

- 17** A decimal number
\$ The last line of the file
/pat/ Search forward for line containing *pat*
\pat Search backward for line containing *pat*
line+n *n* lines forward from *line*
line-n *n* lines backward from *line*

Defaults:

- ()**
(+1) Use the next line
(1,\$) Use all lines

Commands:

- () a** Append text after line (text follows)
() c Change text (text follows)
() d Delete text
() i Insert text before line (text follows)
() p Print text (can be appended to other commands)
() r file Read *file*, appending after line
() s/pat/new/gp Substitute *new* for leftmost *pat* (**g** implies all occurrences)
(1,\$) w file Write *file*, leave current text unaltered (if no file is specified, write to current filename)
() =[p] Print line number, current line

Arguments:

\$n Write to/read from the *n*th temporary buffer

EXAMPLES

```
$ edstr -e s/joe/mary/g -f rfil infile > outfile
```

where *rfil* is a file of one line:

```
20r add_stuff
```

This command first replaces all occurrences of *joe* with *mary*, then copies material in the file *add_stuff* into *infile* following line 20. Results are written to the file *outfile*.

SEE ALSO

More information is available. Type

help edstr commands For a summary of *edstr* commands

help ed commands For a complete description of the commands

NAME

em3270 – emulate an IBM 3270 terminal

SYNOPSIS

em3270.{device}

DESCRIPTION

em3270 allows a Domain node to emulate an IBM 3270 terminal over a serial I/O (SIO) line connected to a VT100-to-3270 converter. The command is meaningless without this additional hardware.

While **em3270** requires no arguments or options, there are actually three different commands, depending on which protocol converter you use. The following protocol converters support the **em3270** package software:

- ICCI Model CA20
- ICCI Model CA12
- KMW Model BAC-3270 FS
- PCI 1076

Specify the device name with the **em3270** command. For example,

```
$ em3270.pci
```

if you are using the PCI 1076 protocol converter.

Follow the manufacturer's directions for connecting the converter you choose to the node's SIO lines.

Once you have invoked **em3270**, you may use the following commands:

- | | |
|-----------------|--|
| h | Display command summary information. |
| li n | Select SIO line <i>n</i> . The default SIO line is 1. |
| q | Exit from em3270 . |
| speed n | Set SIO line speed. Valid speeds are 50, 75, 110, 134, 150, 300, 600, 1200, 2000, 2400, 3600, 4800, 7200, 9600, and 19200. |
| [no]sync | Enable/disable XON/XOFF on the SIO line. |

In addition to these commands, two control-key sequences perform special functions:

- | | |
|----------------|---|
| CTRL/F8 | Switch between command mode and remote 3270 mode. |
| CTRL/F7 | Display a layout of the 3270 emulation keyboard. |

KEYBOARD CONVERSION

The following special keyboard keys map to the IBM equivalents indicated.

Hex Code	IBM Keyboard	Apollo Keyboard
X'5F'	CENT SIGN	LEFT BRACKET '['
X'4A'	NOT SIGN (PLI-NOT)	RIGHT BRACKET ']'
X'6A'	DOUBLE VERTICAL BAR (ONE ABOVE THE OTHER)	CARET '^'
X'4F'	VERTICAL BAR (PLI-OR)	DOUBLE VERTICAL BAR

SEE ALSO

More information is available. Type

help em3270 commands For the above list of em3270 commands

help vt100 For details about emulating a VT100 terminal

NAME

emt – emulate a dumb terminal

SYNOPSIS

emt [*pathname*]

DESCRIPTION

emt allows your node to emulate an ASCII terminal connected to another computer. This asynchronous connection exists through a stream opened on one of the node's SIO lines. **emt** also permits ASCII file transfer between your node and the remote host.

pathname (optional) Specify file containing **emt** commands.

Default if omitted: read commands from standard input

emt begins execution in local mode, and displays the following prompt:

emt>

To enter remote mode, press F1. (The **emt** command **dl** no longer exists.) In remote mode, your terminal operates as if it were physically connected to the remote computer ("host"). You can log on and enter remote host commands.

To return to local mode, press F1 again.

INPUT/OUTPUT STREAMS

emt uses the four standard streams: standard input, standard output, error input, and error output, as follows:

- **emt** commands are read from an **emt** command file or from standard input. The command filename may be specified on the command line or using the **emt run** command. Up to four levels of command files may be nested. When EOF is reached in a command file, commands are read from the previous file or from standard input. If EOF is reached on standard input, **emt** exits.
- Keystrokes to be sent to the host computer are read from standard input only.
- The **emt** command responses and all messages from the host are written to standard output.
- Error messages from Aegis system calls are written to error output. Optional monitoring (**monit**) may also be written to error output (or to a named file).

You may use redirection of standard input, command-line specification of a command file or the **emt run** command to automate **emt** usage and use **emt** in shell scripts. **emt** behaves slightly differently with regard to host transmissions, depending on which of these techniques you use and you may select the method that best suits your purpose.

When input is redirected to standard input ('**emt <emtfile1**'), lines in the command file that are sandwiched between F1 commands (enter/exit remote mode) are transmitted to

the host. Other lines outside F1 commands are interpreted and executed as `emt` commands.

Contents of `emtfile1`:

Command	Description
<code>interm lf</code>	Sets input terminator.
<code>outterm lf</code>	Sets output terminator.
<code>list</code>	Lists <code>emt</code> state settings.
<code>F1</code>	Invokes remote mode (communication to host).
<code>hello host</code>	This and succeeding lines get sent to host.
<code>goodbye host</code>	Last line sent to host.
<code>~li</code>	<code>emtesc</code> char, specifies 'F1', return to local mode.
<code>list</code>	Back in local mode, lists <code>emt</code> state settings.
<code>q</code>	Exit from <code>emt</code> .

When a command file is invoked either via the command line (`emt emtfile2`) or by using the `run` command (`run emtfile2`), the behavior is different in that lines following F1 commands are not transmitted to the host. This is because host transmissions are read from standard input and standard input has not been redirected to the file:

Contents of `emtfile2`:

Command	Description
<code>interm lf</code>	Sets input terminator.
<code>outterm lf</code>	Sets output terminator.
<code>list</code>	Lists <code>emt</code> state settings.
<code>F1</code>	Invokes remote mode (communication to host). All host input is now taken from the keyboard (or from standard input if it has been otherwise redirected). Finally user types <code>~1</code> or presses <code>F1</code> to return to local mode.
<code>list</code>	Local mode, <code>emt</code> commands read from <code>emtfile2</code> again.
<code>q</code>	Exit from <code>emt</code> .

You may also use the `xmit` command to transmit a file (of commands or data) to the host. Use the `emt rcv` command to receive host transmissions to a Domain file.

TRANSFERRING FILES

You can transfer files using **emt**'s receive (**rcv**) or transmit (**xmit**) commands. **xmit** sends a Domain file to the remote host. **rcv** opens a Domain file to receive information from the remote host. For example, if you type (in local mode)

```
emt> xmit fileA
```

emt displays the following message:

```
Ready to transmit file fileA
```

Next, press **F1**. **emt** enters remote mode, and transmits **fileA** to the remote host.

If you type:

```
emt> rcv fileB
```

emt displays this message:

```
Ready to receive file fileB.
```

Next, enter remote mode by pressing **F1**. Use a remote host command to display the information that you want **fileB** to receive. **emt** automatically writes this and all subsequent host transmissions into **fileB**. To stop the **rcv**, press **F2**.

TRANSMISSION CONVENTIONS

Use the **emt** command **interm** to specify the line terminator used by the host. If you do not know what the host uses as a line terminator, experiment by changing **interm**. Use the **emt** command **outterm** to specify the line terminator to be transmitted to the host.

emt allows you to open only one Domain file at a time. If **emt** receives a **xmit** or **rcv** command while another Domain file is active, it closes the open Domain file, and executes the new command.

During remote mode, **emt** waits on both the keyboard and SIO line for characters to process, and monitors the data for characters of special interest to **emt**.

You can specify which keyboard characters **emt** should interpret by placing the keyboard in raw or cooked mode. In raw mode, **emt** passes all keyboard input (except the function keys, keys **L1** through **L12**, and keys **R1** through **R4**), directly to the host. Cooked mode lets you use many of the Display Manager's features for editing the input pad. **emt** places your keyboard in cooked mode by default.

COMMANDS

The following commands are available while running `emt`:

For details about the commands available once `emt` has been invoked, type `help emt commands`

Command	Description
F1	Switch between local and remote modes.
F2	Interrupt a file transfer and close the file.
F3	Turn <code>tee</code> on or off. <code>tee on</code> causes <code>emt</code> to display file transmission records on the screen. You can use this feature to monitor file transfers, and decide if and when you should stop or interrupt a transfer. The default is <code>tee on</code> .
F8	Send a break to the host.
CTRL/F7	Display function key definitions.

These function keys may be simulated by typing the `emt` ESC character followed by the function key number (that is, `~1` for **F1**). When `emt` is used from the VT100 emulator, use `shift F1` instead of **F2**, and `CTRL F1` instead of **F3**.

Command	Description
ae	Abort on error.
asconly notasconly	Sift out most non-printing ASCII codes. Eliminates triangles, allows BS, CR, ESC, FF, LF, TAB. The default is <code>notasconly</code> .
break [n]	Set the <code>break</code> duration value to <code>n</code> milliseconds. The default is 200. If set to 0, the F8 (<code>break</code>) key does nothing.
close	Deactivate an <code>rcv</code> file. See the <code>rcv</code> command for related information.
code [xx none]	Set the host-command-code to the hexadecimal number <code>xx</code> . The default is <code>none</code> .
cooked	Place the keyboard in cooked mode. This enables many DM features for editing the input pad, and provides an escape sequence for sending control characters to the remote host. To send the host a CTRL character, precede the character with a tilde (<code>~</code>). The sequence <code>~_</code> transmits a delete character. To send the host a single tilde character, type <code>~</code> .

The `emt` default is cooked mode. Cooked mode always echos keystrokes, so it does not require a full duplex connection to the host. (See the `raw` command for related information.)

Note: The `cooked` and `raw` commands refer only to the transcript pad and keyboard input. The SIO line itself is always in raw mode.

emtesc [*chr*|none]

Set the `emt` escape character to *chr*. Use `none` to disable the escape character. Default is `~` for "cooked" mode, `none` for "raw" mode.

The following three commands are useful when standard input is redirected to a file of `emt` commands:

- f1** Enter remote mode (Simulate function key F1).
- f2** Terminate file transfer (Simulate function key f2).
- f3** Toggle tee mode (Simulate function key F3).
- hangup** Cause modem to break connection with the remote host.
- help** [*tctl*] Display information about `emt` commands or about `tctl` commands.
- line** {1|2|3|*pathname*}
 Select the SIO line. Pathname must specify an SIO device descriptor (for example, `/dev/sio2`). The default SIO line is 1 (`/dev/sio1`).
- l** Display the current SIO line, all `emt` switch settings and the receive filename, if any.
- monit** [*pathname*]
 Write every character received over the SIO line to *pathname*. If a filename is not specified, the previous specification or error output is used.
- nomonit** Stop monitoring.
- quit** End the `emt` session.
- raw** [-*echo*|-*noecho*] [-*lf*|-*nolf*]
 Place the keyboard in "raw" mode. This sends keyboard input directly to the remote host, interpreting only function keys. The `-echo` option echos keystrokes on standard output; you should use it when the host is in half-duplex mode. The default is `-noecho`. The `-lf` option converts carriage return (CR) to line feed (LF) for lines echoed. The default is `-nolf`. (See the `cooked` command for related information.) Note: The `-echo` and `-lf` options are purely local functions that enable you to read what you type. They do not in any way change host/node transmissions.

rcv [-r] [-keys|-nokeys] [*pathname*]

Prepare the Domain file specified to receive remote host transmissions. If *pathname* already exists, **emt** appends the transmission to it, unless you specify **-r**. The receive begins when you enter remote mode F1. If you omit the *pathname*, **emt** uses the previous name, if any. The **-keys** option writes keystrokes to the file along with received data. The default is **-nokeys**.

emt allows you to interrupt an **rcv** command at any time by pressing F2. **emt** remains in whatever mode it was in, but keeps the **rcv** file active. When you are ready to continue receiving host transmissions, you may type **rcv** again (in local mode) without a filename, and **emt** uses the same **rcv** file.

If you omit filename and no **rcv** file is active, **emt** issues an error message. If you specify a new **rcv** file while another **rcv** file is active, **rcv** closes the active file, and prepares the new file to receive the transmission.

Use the **close** command to deactivate an **rcv** file.

tctl {*tctl commands*}

If you are running under Aegis, pass this command line to the shell command **tctl** to configure the SIO line. If this SIO line is not the default line, then you must use the **-line** command. The **speed** and **sync** commands have been superseded by this direct invocation of **tctl**. If only UNIX is installed, use **stty** to perform this action. If both UNIX and Aegis are installed, you can use either **tctl** or **stty**.

stty See **tctl**.

interm {*cr|lf|crlf|vax|'hex'*}

Select the input line terminator. The default is **crlf**.

outterm {*cr|lf|crlf|'hex'*}

Select the output line terminator. The default is **cr**. **emt** transmits the selected hexadecimal value as the terminator for each line.

xmit *pathname*

Prepare to transmit the Domain file specified to the remote host. If you omit *pathname*, or if you specify a file that does not exist, **emt** issues an error message. When you issue this command, **emt** remains in local mode. **emt** transmits the file when you press F1.

When `emt` completes the transfer, it closes the file and returns to the previous mode. `emt` does not send an end-of-file (EOF) signal to the remote host. If the host requires an EOF, enter remote mode and transmit it manually.

`emt` can also receive commands from the host. If the host transmits the sequence

```
host-command-code (emt command string) line-terminator
```

`emt` interprets the string as an `emt` command. Use the `emt` command code to define [host-command-code].

Line Terminators	emt Response
<code>crlf</code>	Converts sequence to a line feed, ignoring any null characters that may separate the pair.
<code>cr</code>	Converts sequence to a line feed and ignores LFs.
<code>lf</code>	Interprets it as a line feed, and ignores CRs.
<code>vax</code>	Interprets both CR and CR-LF as terminators and converts them to line feed.
<code>'hex'</code>	Converts the given hexadecimal value to LF.

SEE ALSO

More information is available. Type

`help tctl` For details about configuring an SIO line

NAME

ensubs – enter a protected subsystem

SYNOPSIS

ensubs *subsystem_name*

DESCRIPTION

ensubs is used to enter a protected subsystem at shell command level.

Once in the subsystem, the **subs** command can be used to create new managers for the subsystem or to seal data objects so that only managers of the subsystem can operate on them. Also, subsystem managers can be debugged conveniently in this mode using **debug**, and protected data objects can be examined. Note, however, that access to protected objects requires prior use of the **subs -up** command.

The access control list on the file `/sys/subsys/subsystem_name` determines who can enter the subsystem *subsystem_name*: whoever has read and execute rights to it can enter the subsystem. Usually, this capability should be restricted to the creators of the subsystem or to the system administrator.

ARGUMENTS

subsystem_name (required)

Specify name of subsystem to be entered. The shell searches the directory `/sys/subsys` for the file specified.

SEE ALSO

More information is available. Type

help tctl

For details about configuring an SIO line

NAME

environment – inquire about system environment

SYNOPSIS

`/etc/environment [-c] [-i]`

DESCRIPTION

This command is used by shell scripts to inquire about the current "environment", installed environments, or both.

OPTIONS

If no flags are specified, the current environment is printed. If `-i` is specified, however, and you also want current environment, you must add `-c`.

- `-c` print current environment
- `-i` print installed environments

EXAMPLES

```
$ /etc/environment  
aegis
```

```
$ /etc/environment -c  
aegis
```

```
$ /etc/environment -i  
aegis sysv bsd
```

NAME

eoff – deactivate the shell's **-e** flag

SYNOPSIS

eoff

DESCRIPTION

eoff disables variable evaluation. Variables are evaluated only inside variable expression delimiters, `((expression))`; otherwise, the shell treats the `^var_name` expressions as strings and they are not evaluated. To enable variable evaluation regardless of the context in which the variable appears, specify **eon**.

By default, **eoff** is in effect when a shell is invoked.

If **eoff** is specified in a shell script, it remains in effect until that shell script exits, or until overridden by an **eon** in a nested shell script. When a shell script exits, the variable evaluation is returned to the state in effect just before the script was invoked.

eoff requires no arguments or options.

SEE ALSO

More information is available. Type

help eon	For details about enabling global variable evaluation
help sh	For details about the shell command line interpreter
help shell	For general shell information

NAME

eon – activate the shell's **-e** flag

SYNOPSIS

eon

DESCRIPTION

eon enables variable evaluation regardless of the context in which the variables appear. Normally, variables are evaluated only inside variable expression delimiters, ((*expression*)); otherwise, the shell treats the *var_name* expressions as strings and they are not evaluated.

By default, **eoff** is in effect when a shell is invoked.

If **eon** is turned on in a shell script, it remains on until that shell script exits, or until overridden by an **eoff** in a nested shell script. When a shell script exits, the variable evaluation is returned to the state in effect just before the script was invoked.

eon requires no arguments or options.

SEE ALSO

More information is available. Type

help eoff For details about restricting variable evaluation to within variable expressions

help sh For details about the shell command line interpreter

help shell For general shell information

NAME

eqs – compare strings for equality

SYNOPSIS

eqs [*string1* [*string2*]]

DESCRIPTION

eqs compares strings for equality, and sets the abort-severity level accordingly.

ARGUMENTS

If no arguments are specified, eqs always returns true.

string1 (optional) Specify text string to test. If this is the only string given (that is, *string2* is not specified), return true if *string1* is empty; otherwise return false.

Default if omitted: return true

string2 (optional) Specify text string to compare against *string1*. eqs returns true if the strings are equal; false if they are not.

Default if omitted: test *string1* only

EXAMPLES

The following shell script compiles the Pascal module named by the first argument ^1 if the second argument ^2 is -c. Then it binds the module with library.

```
if eqs ^2 '-c' then pas ^1 endif
bind ^1.bin library -b ^1
```

If the second argument is not -c, or if there is no second argument, the program simply binds the module.

NAME

esa – display address of external symbol

SYNOPSIS

esa *symbol_name*

DESCRIPTION

esa displays the address of an external symbol in an installed library. This command is primarily intended for system-level debugging.

symbol_name (required) Specify the symbol whose address you wish to display. **esa** is case sensitive with respect to the symbol name. Lowercase must be used to refer to symbols defined in FORTRAN and Pascal programs. Mixed case may be used, as needed, for symbols defined in C programs.

EXAMPLES

This command displays the address of **gpr_\$init**. This symbol resides within the GPR library, which was installed at system start-up time.

```
$ esa gpr_$init
A1580C
$
```

SEE ALSO

More information is available. Type

help las For information on identification of the library containing the symbol

NAME

exfld – manipulate fields of data

SYNOPSIS

exfld {*field_spec*} *output_format* [*pathname* ...]

DESCRIPTION

exfld manipulates data kept in formatted fields. It copies data from specified fields of the input files to specified places in standard output.

ARGUMENTS

field_spec (required)

Specify either a field list or a free-format separator as follows:

field_list

Integer list identifying fields in the input file to be copied. Up to 9 input fields are allowed. You can specify a field by the columns in which it occurs or by its starting column and length. For example, 5-10 denotes a field that extends from column 5 through column 10, and 3+2 denotes a field that starts in column 3 and spans 2 columns. When specifying more than one field, separate the specifications with commas, for example,

5-10, 16, 72+8

Fields can overlap, and need not be in ascending numerical order. Thus

1-25, 10, 3

is a valid field specification.

-t [*c*]

Free-format separator specification. If input fields do not fall in certain columns, but rather are separated by some character (such as a blank or a comma), describe the fields by using **-t** *c*, replacing *c* with the appropriate separator. A tab character is the default for *c*.

output_format (required)

Specify literal string representing output format. Fields from input are referred to as *\$n* (for example, \$1, \$2, \$3, and so forth) denoting the order the fields are specified in. Up to 9 fields are allowed, plus the argument \$0 which refers to the whole line. Place the *\$n* symbol in the output format wherever the corresponding field should appear, surrounded by any characters desired. For example, an output format specification of

"\$2 somewords \$1"

produces an output line such as

```
field2 somewords field1
```

pathname (optional)

Specify input file to be manipulated.

Default if omitted: read standard input

EXAMPLES

Specify extraction and input text from standard input.

```
$ exfld 1-5,14-18 "$2 follows $1"
```

```
ABCDE is not DEFGH
```

```
DEFGH follows ABCDE
```

```
*** EOF ***
```

```
$
```

NAME

existf – check for existence of an object

SYNOPSIS

existf *pathname* ...

DESCRIPTION

existf reads the object *pathname*(s) you supply and checks to see if the object exists. If the object does exist, existf returns with a good program status (`pgm_>true`). If the object does not exist, existf returns an error status (`pgm_>false`).

ARGUMENTS

pathname required Specify the object to be checked. Multiple pathnames and wild-carding are permitted. If you specify more than one *pathname*, all the objects must exist for existf to return true.

EXAMPLES

Test for `my_file` which does not exist.

```
⚡ if existf my_file then args "The file is there."  
⚡_else args "Out of luck." endif  
Out of luck.  
⚡
```

NAME

existvar – check that a variable is set

SYNOPSIS

existvar *var_name* ...

DESCRIPTION

The **existvar** command checks if the variable name(s) declared as its argument(s) has a currently set value. If the variable is currently set, **existvar** returns a "true" value. If the variable is not currently set, **existvar** returns "false". If you specify more than one variable name to check, all the variables must exist for **existvar** to return "true".

ARGUMENTS

var_name [...] (required) Specify the variable name to be checked. Multiple names are permitted, separated by blanks.

NAME

exit – exit from a loop

SYNOPSIS

exit

DESCRIPTION

exit terminates the flow of control in a shell loop construct (**for**, **select**, and **while**). When **exit** is encountered, control passes to the first command following the body of the loop (see **EXAMPLES** below).

You may also interrupt the flow of control in a loop without actually leaving the loop by using the next command.

Do not confuse this command with the **DM** command **ex**, which exits the Display Manger and returns control to the boot shell. Type **help ex** or see the **ex** command description in the *Domain Display Manager Commands Reference* for more information.

The **exit** command requires no arguments or options.

EXAMPLES

Consider the following section from a shell script:

```
while ((true))
do
  readc a
  if (^a = "y") then exit endif
  args "still looking ..."
enddo
args "Finished."
```

When the **readc** (**read_character**) command reads a character into variable **a** that matches the character **y**, the **exit** command executes and causes the script to jump to the command following the **enddo**.

SEE ALSO

More information is available. Type

help for	For information on for loops
help select	For information on select loops
help while	For information on while loops
help next	For information on next

NAME

export – change a shell variable into an environment variable

SYNOPSIS

export *var_name*...

DESCRIPTION

The shell can access environment variables using all the standard variable commands and operators. The **export** command adds the capability of turning regular shell variables into environment variables.

Environment variables are variables that programs can access or set and that are used to store global state information. Several are generated automatically when you create a process; they can be displayed using the **lvar** (`list_variables`) command. For example,

```
$ lvar
environment NODETYPE = dn400
environment TZ = est5edt
environment PATH = ~/com:/usr/ucb:/bin:/com:/usr/bin
environment TERM = apollo_15P
environment HOME = //node_8e4/joseph
environment USER = joseph
environment LOGNAME = joseph
environment PROJECT = none
environment ORGANIZATION = r_d
environment NODEID = 8E4
$
```

Environment variables are of special interest to users of Domain/OS. Consult the Domain/OS documentation for additional information.

NOTE

The shell creates environment variables in uppercase only. (Environment variables are case sensitive in Domain/OS; the shell allows only uppercase ones to avoid collisions between environment variables and shell variables.)

ARGUMENTS

var_name (required) Specify the shell variable to be changed into an environment variable. It doesn't matter whether the name is typed in uppercase; the shell converts it to uppercase automatically. Multiple variable names are permitted, separated by blanks. If the specified variable does not exist, export creates it.

EXAMPLES

```
$ eon
$ CURRENT_DIR := "//panacea/joe"
$ lvar
string CURRENT_DIR = //panacea/joe
                                (shell variable created.)
environment USER = joe
environment LOGNAME = joe
environment PROJECT = none
environment ORGANIZATION = r_d
environment NODEID = d5b
environment PATH = ~/com:/usr/ucb:/bin:/com:/usr/bin
environment TERM = apollo_191
environment NODETYPE = dn300
environment TZ = est5edt
environment HOME = //panacea/joe
$ export CURRENT_DIR
$ lvar
environment USER = joe
environment LOGNAME = joe
environment PROJECT = none
environment ORGANIZATION = r_d
environment NODEID = d5b
environment PATH = ~/com:/usr/ucb:/bin:/com:/usr/bin
environment TERM = apollo_191
environment NODETYPE = dn300
environment TZ = est5edt
environment HOME = //panacea/joe
environment CURRENT_DIR = //panacea/joe
                                (Environment variable created.)
```


NAME

find_orphans – locate and catalog uncataloged objects

SYNOPSIS

/etc/find_orphans [options] [volume_pathname]

DESCRIPTION

find_orphans finds all uncataloged permanent objects in a local volume. It uses or creates a directory "lost+found" in the root of the volume and creates entries for all objects not cataloged elsewhere.

find_orphans can operate by itself by using search mode, in which case it searches the volume for all orphans, or it works with **salvol** (list mode, the default), in which case it just catalogs the orphans detected by the previous run of **salvol**. Both methods find exactly the same set of orphans. We recommend that you run **find_orphans** every time a node is booted, and on every mounted volume. Below is a description of the two modes of operation.

The objects cataloged by **find_orphans** are given sequential names like f1, f2, etc., and you can move them using `/com/mvf` or `/bin/mv` to a directory of your choice. **find_orphans** is useful for finding objects that were being updated during a system crash or that were uncataloged through program errors.

In list mode (the default), **find_orphans** catalogs all objects listed in the file `lost+found.list` in the root directory of the volume. If the file does not exist, **find_orphans** creates it. Note that **invol** creates the file when it creates the volume. If the `lost+found.list` exists, **salvol** enters information describing each orphan. List mode is considerably faster than search mode since there is no need to search the entire volume. You must have permission to catalog objects in `lost+found`.

In search mode, you must have read permission to all directories on the volume. If some directory is not readable, every object under that directory is cataloged in the `lost+found` directory. In addition, you must have permission either to create the `lost+found` directory or to catalog objects in `lost+found` when it already exists.

Search mode should be run only on a quiescent node; that is, one not connected to the network (use `netsh -n` to disable network communications) and not actively running any processes other than the one performing the `find_orphans` operation.

volume_pathname (optional)

Specify the name of the volume to be searched. The volume must be physically attached to your node; you may not find orphan objects on volumes elsewhere in the network.

Default if omitted: search node boot volume

OPTIONS

- l[ist] (default) List mode, catalog objects listed in `/lost+found.list`.
- s[earch] Search mode, search the volume for orphans.
- v[erify] Verify only; don't catalog any orphans.

EXAMPLES

```
$ /etc/find_orphans
Cataloging in: /lost+found
39D40FF0.100086CA -> f1
39D40F9D.E00086CA -> f2
39D41026.600086CA -> f3
39D40DA6.D00086CA -> f4
39D40998.200086CA -> f5
39D41042.800086CA -> f6
39D40CB8.E00086CA -> f7
39D41001.300086CA -> f8
39D40F7E.D00086CA -> f9
39D40CCE.F00086CA -> f10
39D40D8B.C00086CA -> f11
39D40E33.100086CA -> f12
39D40A06.700086CA -> f13
39D40F23.900086CA -> f14
39D40E16.000086CA -> f15
39D40F36.A00086CA -> f16
39D41C0A.200086CA -> f17
Number of orphans catalogued: 17
```

```
$ ld -a /lost+found
```

```
Directory "/lost+found":
```

sys type	type uid	blocks used	current length	attr	rights	name
file	uasc	1	32	P	prwx-	f1
file	unstruct	0	0	P	prwx-	f10
file	uasc	1	32	P	prwx-	f11
file	unstruct	0	0	P	prwx-	f12
file	unstruct	1	54	P	prwx-	f13
file	uasc	1	32	P	prwx-	f14
file	uasc	1	32	P	prwx-	f15
file	unstruct	0	0	P	prwx-	f16
file	unstruct	0	0	P	prwx-	f17
file	unstruct	0	0	P	prwx-	f2
file	uasc	1	32	P	prwx-	f3
file	unstruct	0	0	P	prwx-	f4
file	coff	1	101376	P	prwx-	f5
file	unstruct	0	0	P	prwx-	f6
file	uasc	1	32	P	prwx-	f7
file	unstruct	0	0	P	prwx-	f8
file	uasc	1	32	P	prwx-	f9

```
17 entries, 9 blocks used.
```

NAME

flen – count lines, words, and characters in a file

SYNOPSIS

flen [*options*] [*pathname* ...]

DESCRIPTION

flen prints the number of lines, words, and characters in each of the named files. A word is defined as any sequence of characters delimited by tabs, spaces, and newlines. If more than one file is specified, totals for all the files are printed also.

ARGUMENTS

pathname (required) Specify input file. Multiple filenames and wildcarding are permitted.

Default if omitted: read standard input; suppress total counts

OPTIONS

If no options are specified, all counts are reported.

-l Print only line counts.

-w Print only word counts.

-c Print only character counts.

Options may be mixed to achieve the desired reporting results.

EXAMPLES

Print the number of lines and characters in the file **mary**.

```
$ flen -l -c mary
```

NAME

fmc – format text into multiple columns

SYNOPSIS

fmc [*options*] [*pathname* ...]

DESCRIPTION

fmc reads the named files and formats them into multiple columns on standard output. Each input line is placed in one column of an output line; input lines that are longer than the output column width are truncated. This command is useful to format text that is already in the form of a column or list.

ARGUMENTS

pathname (optional) Specify input file. Multiple pathnames are permitted, separated by blanks.

Default if omitted: read standard input

OPTIONS

The options control output format. If no options are specified, the default output format is:

Number of columns	2
Page length	55
Column width	60
Gutter width	8

NOTE

Page length * number of columns must be less than 1200. The total number of characters on a page must be less than 7000.

- c *n*** Specify *n* columns. The default is 2.
- l *n*** Specify page length in *n* lines. **fmc** produces output in pages, but does not place separators between the pages. The default is 55.
- w *n*** Specify column width in *n* characters. Input lines longer than *n* characters are truncated. The default is 60.
- g *n*** Specify gutter width in *n* spaces. The gutter is the space between columns. The default is 8.
- d *n*** Specify display terminal as output device. The column width is set to *n* characters and the page size is set to 24 lines. The number of columns and the gutter width are computed to maximize the amount of information on the screen. The default is 10.

EXAMPLES

This command line first produces a cross-referenced list of all the symbols in the file `sample`, then formats the report in a 3-column list.

```
$ crefs sample | fmc -c 3 -w 22 -g 4
```

NAME

fmt – format a text file

SYNOPSIS

fmt [*pathname* ...] [*options*]

DESCRIPTION

fmt is a general purpose text-formatting program, allowing you to arrange output text according to formatting directives embedded in the input file or typed on standard input.

By default, formatted text is written to standard output. You can use the **-out** option to redirect it to a file.

ARGUMENTS

pathname (optional) Specify input file to be formatted. This argument must precede any command line options. Multiple pathnames and wildcarding are permitted; however, **fmt** concatenates multiple files prior to formatting. If **fmt** cannot find one of the specified input files, control shifts to standard input.

Default if omitted: read standard input

OPTIONS

-f *n* Begin output at the first page numbered *n*.
-t *n* Terminate output at the first page numbered higher than *n*.
-s Stop before printing each page, including the first. This option is useful for paper manipulation. The prompt "Type return to begin a page" is issued only once, before the first page.
-po *n* Page offset. Shift the entire document *n* spaces to the right.
-lf List names of files as they are processed.
-out *pathname* Specify output file. If this option is omitted, formatted text is written to standard output.

EXAMPLES

Format **mary** with a page offset of 9 spaces, and write the results to **mary.formatted**.

```
$ fmt mary -out mary.formatted -po 9
```

SEE ALSO

More information is available. Type

help fmt commands For a summary of **fmt** formatting directives

NAME

for – execute a for statement

SYNOPSIS

```
for var_name := int_expr [to int_expr] [by int_expr] command... endfor
for var_name in string_expr [by {char|word|line}] command... endfor
```

DESCRIPTION

for allows you to build a control structure that executes commands repeatedly as long as the result of a Boolean test is true. The **for** command has two formats: one for assigning and testing integer expressions; the other for assigning and testing string expressions.

In the integer form, the (optional) **to** and **by** clauses permit you to specify ranges and increment values, respectively. For example, you might want to loop five times by specifying

```
for a := 0 to 10 by 2
```

If you do not specify **by** *int_expr*, the default increment is 1. If you do not specify **to** *int_expr*, you probably want to increment the variable manually inside the body of the loop. You should also put a test condition inside the loop (and probably use an **exit** to get out) or you risk looping forever.

In the string form, the (optional) **by** clause allows you to control the string assignment operation. If you specify **by** *word* (the default), each word (a sequence of non-blank characters) in *string_expr* is assigned to *var_name* one at a time until *string_expr* is exhausted. You may also assign string values a character at a time, or a line at a time, by using the **by char** and **by line** clauses, respectively.

ARGUMENTS

- var_name* (required) Specify the name of the shell variable whose value is to be assigned and tested.
- int_expr* (required) Specify any valid expression that returns an integer value.
- string_expr* (required) Specify any valid expression that returns a string value.
- command...* (required) Specify the command to be executed as long as the **for** test returns true. This may be a shell command, a shell script, a variable assignment, or any other valid shell operation. Multiple commands are permitted; separate them with semicolons or newline characters.

EXAMPLES

The following example demonstrates the advantages of a for loop over a while loop in one instance. Assume these lines appear in a shell script.

```
# A loop using while.
eon
a := 0
while ((^a <= 10)) do
    args ^a
    a := ^a + 2
enddo
#
# The same loop using for.
#
for a := 0 to 10 by 2
    args ^a
endfor
#
# End of script.
```

This example assigns three names to a variable.

```
#
# Script file_name
#
eon
for file in "foo bar zap" by word
    args ^file
endfor
#
# end of script.
```

Execution produces this:

```
$ file_name
  foo
  bar
  zap
$
```

NAME

fpat – find a text pattern in an ASCII file

SYNOPSIS

fpat [*options*] [*pathname... -p*] *reg_expr* ...

DESCRIPTION

fpat searches its input file(s) for lines matching the specified regular expressions and writes them to standard output or the file specified.

ARGUMENTS

reg_expr... (required) One or more regular expression patterns. By default, a line that contains any of these expressions matches and is written to standard output. For a description of regular expressions used for pattern matching, type **help patterns**. Patterns containing embedded spaces or shell special characters must be enclosed in quotation marks.

pathname -p (optional) Specify name of file to be searched. If you specify a pathname with this argument, you must follow it with **-p** to separate the pathname(s) from the search patterns on the command line. Multiple pathnames and wildcarding are permitted.

Default if omitted: read standard input

OPTIONS

If no options are specified, any line that matches any of the regular expressions is considered a matching line.

-out *pathname* Write output to specified file. If input filenames are specified, the output filename can be derived. If this option is not specified, matching lines are written to standard output.

-a Select only lines that match all regular expressions, in any order.

-x Select only lines containing none of the regular expressions.

-c Write only a count of matching lines, not the lines themselves.

-i Ignore cases for search (that is, become case-insensitive).

-l Write line number with each line that matches the regular expression.

-m *n* Set the maximum number of search lines to *n* (a decimal value). **fpat** terminates after searching *n* lines.

-lf Display the name of the file being examined before searching its lines.

- lm** Similar to **-lf**, but display the name(s) of only those file(s) that contain matches for the regular expression.
- rm *n*** Set the maximum number of matches to be reported for this execution of **fpat**.
- rmf *n*** Set the maximum number of matches to be reported for each file being searched.

EXAMPLES

Assume the file **text** contains the following:

```
now
is
the
time
for
all
good
```

Then the command

```
$ fpat text -p o
```

produces ...

```
now
for
good
$
```

... and the command

```
$ fpat -x -m 5 -l text -p o
```

produces ...

```
( 2) is
( 3) the
( 4) time
( 6) all
$
```

Search for the string "the" in all files whose names begin with "text".

```
$ fpat text?* -p the
```

Search for the string /fBthe in all files whose names begin with text, (that is, text, text1, text_file, etc.) and write the output to the files text.out, text1.out, text_file.out, etc.

```
$ fpat text?* -p the -out =.out
```

SEE ALSO

More information is available. Type

help fpatb For details about searching for blocks of lines containing text patterns

help patterns For a description of regular expressions

NAME

fpatb – find blocks of text containing patterns

SYNOPSIS

fpatb [*options*] [*pathname... -p*] *reg_expr* ... [*-out pathname*]

DESCRIPTION

fpatb reads blocks of text from its input files and writes them to its output file(s) so that they meet the specified matching criteria. By default, blocks of lines are separated by an empty line or by a line containing only blanks. **fpatb** is similar to **fpat** (**find_pattern**) except that if a pattern is found, the entire block of lines is copied to output, rather than only the line in which the pattern occurs. Thus, it is useful for searching mailing lists, bibliographies, and similar files, where several lines are grouped together to form cohesive units.

ARGUMENTS

reg_expr (required) Specify the regular expression to be used for matching search. Each expression defines a text pattern, and you can specify up to nine expressions with each **fpatb** command. **fpatb** is case-sensitive; for example, "a" is different from "A". For a description of regular expressions used for pattern matching, type **help patterns**.

pathname -p (optional)

Specify the name of the file to be searched. If you specify a pathname with this argument, you must follow it with **-p** to separate the pathname(s) from the search patterns on the command line. Multiple pathnames and wildcarding are permitted. Blocks must be less than 15,000 characters.

Default if omitted: read standard input

OPTIONS

If no options are specified, any block containing a line that matches any one of the regular expressions is considered a matching block.

-a Select only blocks containing lines that match all regular expressions, in any order.

-x Select only blocks containing none of the regular expressions.

-c Write only a count of matching lines, not the lines themselves.

-b *reg_expr1* Specify *reg_expr1* as the block separator, instead of a blank or empty line. Text blocks begin at lines containing *reg_expr1*. If **-b** is specified and **-e** is not, *reg_expr1* begins and ends the text block.

- e *reg_expr2*** Specify *reg_expr1* to start a block and *reg_expr2* to end a block. Note that the **-e** option is used only in conjunction with the **-b** option.
- l *n*** Write only the first *n* lines of selected blocks. If a block contains fewer than *n* lines, this option pads the output block with blank lines.
- lf** Display the name of the file being examined before searching its lines.

EXAMPLES

```
$ fpatb address_list -p 01824 -out zip_list
$
```

Locate text blocks with the string **01824** in the file `address_list` and write the results to `zip_list`.

SEE ALSO

More information is available. Type

- help fpat** For details about searching files for single lines containing text patterns
- help patterns** For a description of regular expressions

NAME

french_to_iso – convert files to ISO format

SYNOPSIS

french_to_iso *input_file output_file*

DESCRIPTION

These utilities convert files written with the overloaded 7-bit national fonts to the International Standards Organization (ISO) 8-bit format. The overloaded fonts include any with a specific language suffix (for example, *f7x13.french*, or *din_f7x11.german*). If you created and/or edited a file using one of the national fonts, that file is a candidate for conversion.

You are not required to convert files, but probably will want to because

1. The overloaded fonts replace certain ASCII characters with national ones, have that subset of ASCII characters and the national characters in one file. The 8-bit fonts available as of SR10 include all the ASCII characters and the national characters.
2. The 8-bit fonts also include a wider range of national characters, so you can enter any character in any western European language. This was not always possible with the overloaded fonts. For example, there was not enough space in the overloaded font to include all the French characters, but they all exist in the 8-bit fonts.

There are two parameters to the conversion utilities. You must provide the name of the file you want to convert (*input_file*) and your *output_file*. If *output_file* already exists, the utilities abort.

The default location for the utilities is */usr/apollo/bin*.

FILES

<i>/usr/apollo/bin/french_to_iso</i>	Converts overloaded French to ISO format
<i>/usr/apollo/bin/german_to_iso</i>	Converts overloaded German to ISO format
<i>/usr/apollo/bin/nor.dan_to_iso</i>	Converts overloaded Norwegian/Danish to ISO format
<i>/usr/apollo/bin/swedish_to_iso</i>	Converts overloaded Swedish/Finnish to ISO format
<i>/usr/apollo/bin/swiss_to_iso</i>	Converts overloaded Swiss to ISO format
<i>/usr/apollo/bin/uk_to_iso</i>	Converts overloaded U.K. English to ISO format

DIAGNOSTICS

All messages are generally self-explanatory.

NAME

fserr – find spelling errors

SYNOPSIS

fserr [*pathname ...*] [*options*]

DESCRIPTION

fserr copies the named files line-by-line to standard output, while looking up each word in a dictionary. If it finds any spelling errors on a line, or if it finds words that are not in the spelling dictionary, **fserr** prints the line containing the questionable word and asks if the word is spelled correctly. If you indicate that the word is misspelled, **fserr** prompts for the correct spelling. **fserr** corrects the spelling on standard output and continues.

fserr uses three ASCII files. The large standard dictionary file is `/sys/dict`, which contains the bulk of the words known to **fserr**. Add words to this file if you want them to become permanent additions to your dictionary, making sure entries remain in alphabetical order. (Use the `srf` (`sort_file`) command to alphabetize the file if necessary.) If you do not wish to alter the standard dictionary, you may direct **fserr** to use a file containing your own special words by specifying the `-d` option each time you invoke the command.

`/sys/dictdx` serves as an index into the large dictionary file to speed searches. Do not edit this file manually. If you change `/sys/dict`, delete the index file; **fserr** generates a new one if `/sys/dictdx` does not exist. Note that it takes some time to generate this index, so be prepared for a delay the first time you use **fserr** after changing the dictionary.

Finally, a relatively few common words that occur with great frequency are stored in `/sys/cdict`. These are read and put into an internal hash table each time **fserr** starts up, making access to them faster than looking in the large dictionary file. This list of words is not alphabetized; rather, words appear in order of relative frequency, with the most common words at the top of the file. You may change this file if necessary. Just be careful not to make the file too big, since that would defeat the purpose of a quick lookup for common words.

ARGUMENTS

pathname (optional) Specify the file containing text to be checked. Multiple pathnames are permitted, separated by blanks.

Default if omitted: read standard input

OPTIONS

- f** Process words just after a period ('.') in column 1 (that is, fmt directives). The default is to ignore such words.
- n** Process digits. The default is to ignore digits.
- u** Underline misspelled words instead of prompting for correction or verification.
- s** Collect and print statistics on dictionary use.
- c *pathname*** Write words that are not in the dictionary, but are correctly spelled, into *pathname*.
- d *pathname*** Add the words in the file *pathname* to the dictionary used for this run. Words in the file must appear one per line.

NAME

fst – print fault status information

SYNOPSIS

fst [[-s] [-r] | [-a]] [-u *n*]

DESCRIPTION

fst prints information about the most recent fault that occurred in the process. The information always includes the fault status, the program counter (PC) at which the fault occurred, and a textual description of the error as reported by the system call `error_$print`. **fst** is intended for system-level debugging.

If you are using a Peripheral Bus Unit (PBU) device, you can get fault information by using the `-u` option (see below).

fst is obsolete and is valid only when running in INPROCESS compatibility mode with the `inprocess` variable set and all commands running in-process. Use the command `tb -full` instead of **fst**.

OPTIONS

- `-r` Print the contents of the CPU general registers when the fault occurred.
- `-s` Print the supervisor PC, entry control block (ECB), and status register (SR) if the fault occurred in supervisor mode. This option is ignored if the fault occurred in user mode.
- `-a` Print all available fault information. (Prints the same information as both `-s` and `-r`.)
- `-u n` Print the same information as both `-s` and `-r` for faults caused by the PBU interrupt handler for unit *n*.

EXAMPLES

```
$ fst -a
```

```
Fault Diagnostic Information
Fault Status = 00120010:
process quit (from OS / fault handler)
User Fault PC = 000157FC
D0-D7: 00120010 00000000 00000002 FFFFFFFE 00000008 00000006 \
00000182 00000004
A0-A7: 0020A452 00E2F22E 0020A3D4 0020A450 00E2F174 0000C92C \
002746B4 002746AC
Supervisor ECB = 00000000
Supervisor SR = 0000
Supervisor PC = 00000000
```

NAME

ftp – ARPANET file transfer program

SYNOPSIS

ftp [*-v*] [*-d*] [*-i*] [*-n*] [*-g*] [*host*]

DESCRIPTION

ftp is the user interface to the ARPANET standard File Transfer Protocol (FTP). The program allows you to transfer files to and from a remote network site.

You can specify the client host with which **ftp** is to communicate on the command line. If you do, **ftp** immediately attempts to establish a connection to an FTP server on that host; otherwise, **ftp** enters its command interpreter and awaits instructions from you. When **ftp** is awaiting commands from you, it displays the prompt “**ftp>**”.

COMMANDS

ftp: recognizes the following commands:

! [*command*] [*args*]

Invoke an interactive shell on the local machine. If you specify arguments, **ftp** takes the first to be a command to execute directly, with the rest of the arguments as its arguments.

\$ *macro-name* [*args*]

Execute the macro *macro-name* that was defined with the **macdef** command. Arguments are passed to the macro unglobbed.

account [*passwd*]

Supply a supplemental password required by a remote system for access to resources once a login has been successfully completed. If you do not specify an argument, **ftp** prompts you for an account password in a non-echoing input mode.

append *local-file* [*remote-file*]

Append a local file to a file on the remote machine. If you do not specify *remote-file*, **ftp** uses the local filename, after applying the changes required by any **ntrans** or **nmap** setting, to name the remote file. **ftp** uses the current settings for **type**, **form**, **mode**, and **structure**.

ascii

Set the file transfer type to network ASCII. This is the default type.

bell

Arrange that a bell be sounded after each file transfer command is completed.

binary

Set the file transfer type to support binary image transfer.

bye

Terminate the FTP session with the remote server and exit **ftp**. An end-of-file also terminates the session and exits.

case

Toggle remote computer filename case-mapping during **mget** commands. When **case** is on (the default is off), remote computer filenames with all letters in uppercase are written in the local directory with the

letters mapped to lowercase.

cd *remote-directory*

Change the working directory on the remote machine to *remote-directory*.

cdup

Change the remote-machine working directory to the parent of the current remote-machine working directory.

close

Terminate the FTP session with the remote server, and return to the command interpreter. Any defined macros are erased.

cr

Toggle carriage-return stripping during ASCII-type file retrieval. Records are denoted by a carriage-return/linefeed sequence during ASCII-type file transfer. When **cr** is on (the default), carriage returns are stripped from this sequence to conform with the UNIX single-linefeed record delimiter. Records on non-UNIX remote systems may contain single linefeeds; when an ASCII-type transfer is made, you can distinguish these linefeeds from a record delimiter only when **cr** is off.

delete *remote-file*

Delete the file *remote-file* on the remote machine.

debug [*debug-value*]

Toggle debugging mode. If you specify an optional *debug-value*, **ftp** uses it to set the debugging level. When debugging is on, **ftp** prints each command sent to the remote machine, preceded by the string "-->".

dir [*remote-directory*] [*local-file*]

Print a listing of the directory contents in the directory, *remote-directory*, and, optionally, place the output in *local-file*. If you do not specify a directory, **ftp** uses the current working directory on the remote machine. If you do not specify a local file, or *local-file* is -, **ftp** sends output to the terminal.

disconnect

A synonym for **close**.

form *format*

Set the file transfer form to *format*. The default and only supported format is **file**.

get *remote-file* [*local-file*]

Retrieve the *remote-file* and store it on the local machine. If you do not specify the local filename, **ftp** gives it the same name it has on the remote machine, subject to alteration by the current **case**, **ntrans**, and **nmap** settings. **ftp** uses the current settings for **type**, **form**, **mode**, and **structure** while transferring the file.

glob

Toggle filename expansion for **mdelete**, **mget** and **mput**. If you turn globbing off with **glob**, **ftp** takes the filename arguments literally and does not expand them. Globbing for **mput** is done as in **cs(1)**. For **mdelete** and **mget**, each remote filename is expanded separately on the

remote machine and the lists are not merged. Expansion of a directory name is likely to be different from expansion of an ordinary filename: the exact result depends on the foreign operating system and FTP server. You can preview the results by executing '*mlls remote-files -*'. Note: *mget* and *mput* are not meant to transfer entire directory subtrees of files. You can do that by transferring a *tar(1)* archive of the subtree (in binary mode).

hash Toggle hash-sign (#) printing for each data block transferred. The size of a data block is 1024 bytes.

help [*command*]

Print an informative message about the meaning of *command*. If you do not specify an argument, *ftp* prints a list of the known commands.

lcd [*directory*]

Change the working directory on the local machine. If you do not specify a *directory*, *ftp* uses your home directory.

ls [*remote-directory*] [*local-file*]

Print an abbreviated listing of the contents of a directory on the remote machine. If you do not specify *remote-directory*, *ftp* uses the current working directory. If you do not specify a local file, or if *local-file* is *-*, *ftp* sends the output to the terminal.

macrodef *macro-name*

Define a macro. Subsequent lines are stored as the macro *macro-name*; a null line (consecutive newline characters in a file or carriage returns from the terminal) terminates macro input mode. There is a limit of 16 macros and 4096 total characters in all defined macros. Macros remain defined until you execute a close command. The macro processor interprets '\$' and '\' as special characters. A '\$' followed by a number (or numbers) is replaced by the corresponding argument on the macro-invocation command line. A '\$' followed by an 'i' signals that macro processor that the executing macro is to be looped. On the first pass '\$i' is replaced by the first argument on the macro-invocation command line, on the second pass it is replaced by the second argument, and so on. A '\' followed by any character is replaced by that character. Use the '\' to prevent special treatment of the '\$'.

mdelete [*remote-files*]

Delete the *remote-files* on the remote machine.

mdir *remote-files local-file*

This command works like *dir*, except that you can specify multiple remote files. If interactive prompting is on, *ftp* prompts you to verify that the last argument is indeed the target local file for receiving *mdir* output.

mget *remote-files*

Expand the *remote-files* on the remote machine and execute a **get** for each filename thus produced. See **glob** for details on the filename expansion. Resulting filenames are then processed according to case, **ntrans**, and **nmap** settings. Files are transferred into the local working directory, which you can change with '**lcd** *directory*'; You can create new local directories with '**! mkdir** *directory*'.

mkdir *directory-name*

Make a directory on the remote machine.

mls *remote-files local-file*

This command is like **ls**, except that you can specify multiple remote files. If interactive prompting is on, **ftp** prompts you to verify that the last argument is indeed the target local file for receiving **mls** output.

mode [*mode-name*]

Set the file transfer mode to *mode-name*. The default and only supported *mode-name* is **stream**.

mput *local-files*

Expand wildcards in the list of local files given as arguments and execute a **put** for each file in the resulting list. See **glob** for details of filename expansion. Resulting filenames are then processed according to **ntrans** and **nmap** settings.

nmap [*inpattern outpattern*]

Set or unset the filename-mapping mechanism. If you do not specify an argument, the filename-mapping mechanism is unset. If you specify an argument, **nmap** maps remote filenames during **mput** commands and **put** commands issued without a specified remote-target filename. and maps local filenames during **mget** commands and **get** commands issued without a specified local-target filename. This command is useful when you are connecting to a non-UNIX remote computer with different file-naming conventions or practices.

The mapping follows the pattern set by *inpattern* and *outpattern*. *Inpattern* is a template for incoming filenames (which may have already been processed according to the **ntrans** and case settings). Include the sequences '\$1', '\$2', ..., '\$9' in *inpattern*, if you want variable templating. Use '\\$' to prevent this special treatment of the '\$' character. **nmap** treats all other characters literally, and uses them to determine the **nmap** *inpattern* variable values. For example, given *inpattern* \$1.\$2 and the remote filename "mydata.data", \$1 has the value "mydata", and \$2 has the value "data".

The *outpattern* determines the resulting mapped filename. The sequences '\$1', '\$2', ..., '\$9' are replaced by any value resulting from the *inpattern* template. The sequence '\$0' is replaced by the original filename. Additionally, the sequence '[*seq1,seq2*]' is replaced by *seq1* if *seq1* is not a null string; otherwise it is replaced by *seq2*. For example, the command `nmap $1.$2.$3 [$1,$2].[$2,file]` yields the output filename `myfile.data` for input filenames `myfile.data` and `myfile.data.old`, `myfile.file` for the input filename `myfile`; and `myfile.myfile` for the input filename `.myfile`. You can include spaces in *outpattern*, as in the example: `nmap $1 |sed "s/ */ /" > $1`. Use the backslash character to prevent special treatment of the '\$', '[', ']', and ',' characters.

ntrans [*inchars* [*outchars*]]

Set or unset the filename-character-translation mechanism. If you do not specify an argument, the filename-character-translation mechanism is unset. If you specify an argument, `ntrans` translates characters in remote filenames during `mput` commands and `put` commands issued without a specified remote-target filename, and translates characters in local filenames during `mget` commands and `get` commands issued without a specified local-target filename.

This command is useful when you are connecting to a non-UNIX remote computer with different file-naming conventions or practices. `ntrans` replaces characters in a filename matching a character in *inchars* with the corresponding character in *outchars*. If the character's position in *inchars* is longer than the length of *outchars*, `ntrans` deletes the character from the filename.

open host [*port*]

Establish a connection to the specified *host* FTP server. You can specify an optional port number, in which case `ftp` attempts to contact an FTP server at that port. If the auto-login option is on (default), `ftp` also attempts to automatically log you in to the FTP server (see below).

prompt

Toggle interactive prompting. Interactive prompting occurs during multiple file transfers to allow you to selectively retrieve or store files. If prompting is turned off (default is on), any `mget` or `mput` transfers all files, and any `mdelete` deletes all files.

proxy ftp-command

Execute an `ftp` command on a secondary control connection. This command allows you to connect simultaneously to two remote FTP servers for transferring files between them. The first `proxy` command should be an `open`, to establish the secondary control connection. Enter the command `proxy ?` to see other `ftp` commands executable on the secondary connection. The following commands behave differently when prefaced

by proxy: **open** does not define new macros during the auto-login process, **close** does not erase existing macro definitions, **get** and **mget** transfer files from the host on the primary control connection to the host on the secondary control connection, and **put**, **mput**, and **append** transfer files from the host on the secondary control connection to the host on the primary control connection. Third-party file transfers depend upon support of the FTP protocol PASV command by the server on the secondary control connection.

put *local-file* [*remote-file*]

Store a local file on the remote machine. If you do not specify *remote-file*, **put** uses the local filename after processing according to any **ntrans** or **nmap** settings in naming the remote file. **ftp** uses the current settings for **type**, **form**, **mode**, and **structure**.

pwd Print the name of the current working directory on the remote machine.

quit This is a synonym for **bye**.

quote *arg1 arg2 ...*

This command sends the arguments you specify, verbatim, to the remote FTP server.

recv *remote-file* [*local-file*]

This is a synonym for **get**.

remotehelp [*command-name*]

Request help from the remote FTP server. If a *command-name* is specified it is supplied to the server as well.

rename [*from*] [*to*]

Rename the file *from* on the remote machine, to the file *to*.

reset Clear the reply queue. This command resynchronizes command/reply sequencing with the remote FTP server. Resynchronization may be necessary following a violation of the FTP protocol by the remote server.

rmdir *directory-name*

Delete the specified directory on the remote machine.

runique Toggle storing of files on the local system with unique filenames. If a file already exists with a name equal to the target local filename for a **get** or **mget** command, **runique** appends a **.1** to the name. If the resulting name matches another existing file, **runique** appends a **.2** to the original name. If this process continues up to **.99**, **runique** prints an error message. **ftp** does not execute the transfer, and reports the generated unique filename. Note that **runique** does not affect local files generated from a shell command (see below). The default value is off.

- send *local-file* [*remote-file*]**
This is a synonym for **put**.
- sendport** Toggle the use of PORT commands. By default, **ftp** attempts to use a PORT command when establishing a connection for each data transfer. The use of PORT commands can prevent delays when you perform multiple file transfers. If the PORT command fails, **ftp** uses the default data port. When the use of PORT commands is disabled, **ftp** does not attempt to use PORT commands for each data transfer. This is useful for certain FTP implementations that ignore PORT commands but indicate, incorrectly, that they are accepted.
- status** Show the current status of **ftp**.
- struct [*struct-name*]**
Set the file transfer structure to *struct-name*; either stream or record. By default, **struct** uses stream structure.
- sunique** Toggle storing of files on remote machine under unique filenames. The remote FTP server must support the FTP protocol STOU command for successful completion. The remote server reports unique names. The default value is off.
- tenex** Set the file transfer type to that needed to talk to TENEX machines.
- trace** Toggle packet tracing.
- type [*type-name*]**
Set the file transfer type to *type-name*; one of *ascii*, *binary*, *image*, or *tenex*. If you do not specify a type, **type** prints the current type. The default type is *ascii* (network ASCII).
- user *user-name* [*password*] [*account*]**
Identify yourself to the remote FTP server. If the password is not specified and the server requires it, **ftp** will prompt the user for it (after disabling local echo). If the FTP server requires an account field and you do not specify it, **ftp** prompts for it. If you specify an account field, **ftp** relays an account command to the remote server after the log-in sequence is completed, if the remote server did not require it for logging in. Unless you invoke **ftp** with "auto-login" disabled, **ftp** executes this process automatically, on initial connection to the FTP server.
- verbose** Toggle verbose mode. In verbose mode, **ftp** displays all responses from the FTP server and also reports statistics regarding the efficiency of a file transfer, when the transfer completes. By default, **verbose** is on.
- ? [*command*]**
This is a synonym for **help**.

You can enclose command arguments that have embedded spaces in quotation (") marks.

ABORTING A FILE TRANSFER

Use the terminal interrupt key (usually CTRL/C) to abort a file transfer. `ftp` immediately stops sending transfers. You can stop receiving transfers by sending a FTP protocol ABOR command to the remote server and discarding any further data received. The speed at which this is accomplished depends on the remote server's support for ABOR processing. If the remote server does not support the ABOR command, an "ftp>" prompt does not appear until the remote server has completed sending the requested file.

The terminal interrupt key sequence is ignored when `ftp` has completed any local processing and is awaiting a reply from the remote server. A long delay in this mode may result from the ABOR processing described above, or from unexpected behavior by the remote server, including violations of the FTP protocol. If the delay results from unexpected remote server behavior, you must kill the local `ftp` program by hand.

FILE-NAMING CONVENTIONS

`ftp` processes files that you specify as arguments according to the following rules.

- 1) If you specify the filename as a dash (-), `ftp` uses `stdin` (for reading) or `stdout` (for writing).
- 2) If the first character of the filename is "|", `ftp` interprets the remainder of the argument as a shell command, then forks a shell, using the UNIX `popen` subroutine with the argument you specify, and reads (writes) from `stdout` (`stdin`). If the shell command includes spaces, you must enclose the argument in quotation marks; for example, "'| ls -lt'". A particularly useful example of this mechanism is "dir |more".
- 3) Failing the above checks, if globbing is enabled, `ftp` expands local filenames according to the rules used in the `csh`; See the `glob` command for a comparison. If `ftp` expects a single local file (for example, `put`), it uses only the first filename generated by the "globbing" operation.
- 4) For `mget` commands and `get` commands with unspecified local filenames, the local filename is the remote filename, that a `case`, `ntrans`, or `nmap` setting can change. The remote server can then change the resulting filename, if `runique` is on.
- 5) For `mput` commands and `put` commands with unspecified remote filenames, the remote filename is the local filename, that a `ntrans` or `nmap` setting can change. The remote server can then change the resulting filename, if `sunique` is on.

FILE TRANSFER PARAMETERS

The FTP specification specifies many parameters that may affect a file transfer. The type can be one of `ascii`, `image` (binary), `ebcdic`, and `local` byte size. `ftp` supports the

ascii and image types of file transfer, plus local byte size 8 for tenex mode transfers.

ftp supports only the default values for the remaining file transfer parameters: mode, form, and struct.

OPTIONS

You can specify options on the command line, or to the command interpreter.

- v (verbose on) Forces ftp to show all responses from the remote server, as well as report on data transfer statistics.
- n Restrains ftp from attempting "auto-login" on initial connection. If auto-login is enabled, ftp checks the .netrc (see below) file in your home directory for an entry describing an account on the remote machine. If no entry exists, ftp prompts for the remote machine log-in name (the default is the user identity on the local machine), and, if necessary, prompts for a password and an account with which to log in.
- i Turns off interactive prompting during multiple file transfers.
- d Enables debugging.
- g Disables filename globbing.

THE .netrc FILE

The .netrc file contains log-in and initialization information used by the auto-login process. It resides in your home directory. by spaces, tabs, or newlines:

machine *name*

Identify a remote machine name. The auto-login process searches the .netrc file for a machine token that matches the remote machine specified on the ftp command line or as an open command argument. Once a match is made, the subsequent .netrc tokens are processed, stopping when the end-of-file is reached or another machine token is encountered.

login *name* Identify a user on the remote machine. If this token is present, the auto-login process initiates a login using the specified *name*.

password *string*

Supply a password. If this token is present, the auto-login process supplies the *string* if the remote server requires a password as part of the log-in process. Note that if this token is present in the .netrc file, ftp aborts the auto-login process if the .netrc is readable by anyone besides the user.

account *string*

Supply an additional account password. If this token is present, the auto-login process supplies the *string* if the remote server requires an additional account password, or the auto-login process initiates an ACCT command if it does not.

macdef *name* Define a macro. This token functions like the ftp **macdef** command functions. A macro is defined with the specified *name*; its contents begin with the next **.netrc** line and continue until a null line (consecutive new-line characters) is encountered. If a macro named **init** is defined, ftp automatically executes it as the last step in the auto-login process.

BUGS

Correct execution of many commands depends upon proper behavior by the remote server.

An error in the treatment of carriage returns in the 4.2BSD UNIX ASCII-mode transfer code has been corrected. This correction may result in incorrect transfers of binary files to and from 4.2BSD servers using the ASCII type. Avoid this problem by using the binary image type.

NAME

german_to_iso – convert files to ISO format

SYNOPSIS

german_to_iso *input_file output_file*

DESCRIPTION

These utilities convert files written with the overloaded 7-bit national fonts to the International Standards Organization (ISO) 8-bit format. The overloaded fonts include any with a specific language suffix (for example, *f7x13.french*, or *din_f7x11.german*). If you created and/or edited a file using one of the national fonts, that file is a candidate for conversion.

You are not required to convert files, but probably will want to because

1. The overloaded fonts replace certain ASCII characters with national ones, have that subset of ASCII characters and the national characters in one file. The 8-bit fonts available as of SR10 include all the ASCII characters and the national characters.
2. The 8-bit fonts also include a wider range of national characters, so you can enter any character in any western European language. This was not always possible with the overloaded fonts. For example, there was not enough space in the overloaded font to include all the French characters, but they all exist in the 8-bit fonts.

There are two parameters to the conversion utilities. You must provide the name of the file you want to convert (*input_file*) and your *output_file*. If *output_file* already exists, the utilities abort.

The default location for the utilities is */usr/apollo/bin*.

FILES

<i>/usr/apollo/bin/french_to_iso</i>	Converts overloaded French to ISO format
<i>/usr/apollo/bin/german_to_iso</i>	Converts overloaded German to ISO format
<i>/usr/apollo/bin/nor.dan_to_iso</i>	Converts overloaded Norwegian/Danish to ISO format
<i>/usr/apollo/bin/swedish_to_iso</i>	Converts overloaded Swedish/Finnish to ISO format
<i>/usr/apollo/bin/swiss_to_iso</i>	Converts overloaded Swiss to ISO format
<i>/usr/apollo/bin/uk_to_iso</i>	Converts overloaded U.K. English to ISO format

DIAGNOSTICS

All messages are generally self-explanatory.

NAME

glbd – Global Location Broker Daemon

SYNOPSIS

```
/etc/ncs/glbd [ -create { -first | -from host_name } ]
```

DESCRIPTION

The Global Location Broker Daemon (**glbd**), part of the Network Computing System (NCS), manages the Global Location Broker (GLB) database. The GLB database stores the locations of NCS-based server programs on a network or internet.

The GLB can be replicated for greater availability of its database. Copies of the database can exist on several nodes, with the brokers maintaining consistency of the replicated database. (In an internet, at least one **glbd** must run in each network.) The **drm_admin** tool administers the replication of the GLB database.

Access to the GLB database by clients is supported on both the DARPA IP and the Domain DDS network protocols. However, GLBs use only the DDS protocols to maintain replication of the database. Thus, on an internet, all routing nodes must support DDS.

A Local Location Broker Daemon (**llbd**) must be running on the local node when **glbd** is started. Typically, both brokers are started at boot time from the */etc/rc* file.

If **glbd** is to communicate via IP protocols, a TCP daemon (**tcpd**) must also be running on the local node. **tcpd** should be started before **llbd**.

See *Managing the NCS Location Broker* for more information.

Diagnostic output from **glbd** is written to the file ``node_data/system_logs/glb_log`.

OPTIONS

- create** Create a replica of the GLB. This option creates a GLB database in addition to starting a broker process. It must be used with either **-first** or **-from**.
- first** This option must be used with the **-create** option. Use it to create the first replica (i.e., the very first instance) of the GLB on your network or internet.
- from *host_name*** This option must be used with the **-create** option. Use it to create additional replicas of the GLB. A replica of the GLB must exist at *host_name*. The database and replica list for the new replica are initialized from those at *host_name*. The replica at *host_name* adds an entry for the new replica to its replica list and propagates the entry to the other GLB replicas.

A *host_name* takes the form *family:host*. The only currently supported family is *dds*; a host in this family is specified by its entry directory or its network address. For example, *dds://jeeves* and *dds:#1234.abcd* are acceptable host names.

EXAMPLES

Initialize and start the first replica of the GLB on this network or internet:

```
$ /etc/server/etc/ncs/gldb -create -first &
```

Start a subsequent replica of the GLB, initializing its database from host *//jeeves*:

```
$ /etc/server/etc/ncs/gldb -create -from dds://jeeves &
```

Restart an existing replica of the GLB:

```
$ /etc/server/etc/ncs/gldb &
```

Restart an existing replica of the GLB on remote host *//bertie*:

```
$ crp -on //bertie /etc/server//bertie/etc/ncs/gldb &
```

SEE ALSO

drm_admin, *lb_admin*, *llbd*,
Managing the NCS Location Broker.

NAME

help – provide help on shell and DM commands

SYNOPSIS

help [*topic* [*subtopic*]]

DESCRIPTION

The **help** command provides information on shell and DM commands and miscellaneous system services by opening a window to display the file that you request. For a list of subjects in the help library, type

\$ help index

Access to system help files is also provided through the **HELP** key. This key opens a read-only pad on a help file using your typed input to construct the pathname, so the syntax is slightly different if you are seeking information on a subtopic. In that case, separate the topic and subtopic with a slash (/) instead of a blank. For example:

Help on: **shell/commands**

ARGUMENTS

topic (optional)

Specify the name of the command or topic for which you desire help.

Default if omitted: display introductory information

subtopic (optional)

Specify the subtopic to be viewed. For example,

\$ help shell commands

displays a topical index of shell commands, while

\$ help shell

displays general information about the shell.

Default if omitted: no subtopic displayed.

NAME

hlpver – provide help support for shell scripts

SYNOPSIS

hlpver *script_name* *version* ^1

DESCRIPTION

hlpver provides access to the Domain/OS help system utilities that support the standard command options **-help**, **-version**, and **-usage** for shell commands. By placing the **hlpver** command inside a shell script, you can allow users of the script to specify these three standard command options and receive meaningful output.

hlpver looks for help information in a file called `/sys/help/script_name.hlp`. **help** files have special information at the top that **hlpver** uses. This information must follow a standard format. The following example shows the header of the **help** file for the **cpf** (`copy_file`) command.

```
10.0;wd (working_directory), revision 1.0, 88/06/06
cpf (copy_file) - copy a file
usage:  cpf source_pathname [target_pathname]
        [-c|-r] [-chn] [-f] [-lf] [-ldl]
        [-du] [-dacl|-sacl] [-subs|-nsubs]
        [-pdt] [-cwl] {CL}
```

All **hlpver** output goes to standard output (normally directed to the process transcript pad). **hlpver** returns the first line of the **help** file in response to **-version**. It returns the second line through the first blank line in the file in response to **-usage**. It returns the entire file in response to **-help**.

Any user file placed in the `/sys/help` directory is also available to the **help** command for display in a standard help window. Thus the file `/sys/help/mary.hlp` can be viewed with

```
$ help mary
```

regardless of whether you are using **hlpver** inside the script **mary**. **hlpver** is solely for the purpose of enabling the three standard command options mentioned above.

ARGUMENTS

script_name (required) Specify the name of the script for which help is to be provided. The name is the right-most leaf in the pathname, not the entire pathname of the script. **hlpver** uses this name to construct the pathname for the **help** file to be returned (that is, `/sys/help/script_name.hlp`).

- version* (required) Specify the version number of the shell script. `hlpver` compares this number to the version number in the help file (the first characters in the file up to the first semicolon) and returns an error if they do not match. This allows you to coordinate versions of the script and the help file.
- `^1` (required) Pass the desired option from the command line. `^1` must appear literally so that `hlpver` can tell what information to return (`-help`, `-version`, or `-usage`). See the example below.

EXAMPLES

Assume that the following lines appear in a file called `test_script`

```
#
# Example script showing hlpver usage.
#
hlpver test_script 1.0 ^1
args "Please enter ..."

# End of script
```

When you type

```
$ test_script -help
```

`hlpver` returns the contents of `/sys/help/test_script.hlp` to the transcript pad. Likewise, when you type:

```
$ test_script -version
```

`hlpver` returns the first line of the help file containing the version number.

NAME**hpc** – program counter histogram**SYNOPSIS**

```
hpc [-low x] [-high x]
      [-from procedure]
      [-to procedure]
      [-proc procedure]
      [-limit n] [-rate n]
      [-nhdr] [-map]
      [-brief] pathname
      [args...]
```

DESCRIPTION

hpc (histogram_program_counter), part of Domain/PAK (Domain Performance Analysis Kit), looks at the performance of programs at the PC level.

hpc produces a histogram of the program counter (PC) during program execution, thus helping you locate the most compute-bound portions of your program.

While your program is executing, **hpc** samples the PC at regular intervals, gathering a set of data points. Each data point records the region in which the program was executing the location of the PC when the sample was taken.

hpc divides your program into 256 equally sized regions called "buckets." The size of the region depends on the size of your program or the range you select. The smaller the region, the better the resolution of the analysis.

When execution of your program has ended, **hpc** displays statistics and a histogram (bar graph) of the PC. Each bar corresponds to an area of program memory. The length of the bar indicates how much time the program spent executing in the corresponding area. **hpc** tells you which procedures and line numbers each bar represents.

While **hpc** and your program are executing, a serial line is not available for output.

<i>pathname</i> (required)	Specify the name of the program to be evaluated.
<i>args</i> (optional)	Specify any arguments to be passed to the program <i>pathname</i> . These are not processed by hpc , but passed directly to your program.

Default if omitted: no arguments passed

OPTIONS

If no options are specified, a histogram is produced for the entire program, with 500 samples taken per second.

- low *x*** Specify lowest address *x* to be included in the histogram. *x* must be a hexadecimal value. If this option is omitted, the histogram starts at the beginning of the program or procedure (see **-from** below).
 - high *x*** Specify highest address *x* to be included in the histogram. *x* must be a hexadecimal value. If this option is omitted, the histogram continues to the end of the program or procedure (see **-to** below).
 - from *procedure*** Specify the beginning of a procedure as the lowest address to be included in the histogram. If both **-from** and **-low** are omitted, the histogram starts at the beginning of the program. Note the the procedure name is case-insensitive.
 - to *procedure*** Specify the end of a procedure as the highest address to be included in the histogram. If both **-to** and **-high** are omitted, the histogram stops at the end of the program. Note the the procedure name is case-insensitive.
 - proc *procedure*** Specify a single procedure to be included in the histogram. Note the the procedure name is case-insensitive.
- By limiting the range of addresses in the histogram with **-low**, **-high**, **-from**, **-to**, and **-proc**, you can study a specific part of your program, such as an I/O routine.
- limit *n*** Limit the displayed histogram bars to those that represent more than *n*% of the monitored program execution. The default value for *n* is 1. Use **-limit 0** to show all histogram entries.
 - rate *n*** Specify how many times *n* hpc samples the program counter per second. *n* must be a decimal number in the range 5 to 2000. The default is 500 samples per second. A higher rate results in a more accurate histogram, but tends to slow program execution.
 - nhdr** Generate the histogram without the header information. Using this option makes filtering the output easier.
 - map** Generate a list of the names and starting and ending locations of the procedures in the program. This list is reduced if **-from**, **-to**, **-high**, or **-low** are used to restrict monitoring to specific procedures or memory addresses. The output from this option can be quite verbose for large programs.

-brief

Produce a compact bar chart by showing only the name of the first procedure, or procedure fragment, contained in the bucket represented by each bar. By default, `dpat` shows the names of all procedures or procedure fragments contained in the bucket. This option also suppresses source-line information.

SEE ALSO

More information is available. Type

help dpat

For details about the domain performance analysis tool

help dspst

For details about displaying process status data

NAME

if – execute a conditional statement

SYNOPSIS

```
if condition then  
    command_1 ...  
[else  
    command_2 ...]  
endif
```

DESCRIPTION

if executes a conditional statement depending on the results of a Boolean test. You can extend the **if** command over several lines if you use it interactively or in a shell script. When you use **if** interactively, and extend the command over more than one line, the shell prompts you for each new line of the command with the `$_` prompt (refer to the example below).

ARGUMENTS

condition (required) Specify a command or program to execute and test for truth, or specify a variable expression or Boolean variable to test for truth. "Truth" usually means that the command executes successfully (without any errors), or that the shell variable expression or Boolean is "true". (Specifically, this argument is evaluated true if it returns an abort-severity level of 0 (zero).)

Refer to the manual, *Using Your Aegis Environment* for more information on shell variables.

command_1 (required) Specify command or program to execute if *condition* returns true.

command_2 (optional) Specify command or program to execute if *condition* returns false (that is, a severity level greater than zero).

EXAMPLES

1.

```
$ if eqs a a
  $_ then args "a is a"
  $_ else args "Aristotle was wrong."
  $_ endif
a is a
$
```
2.

```
if eqs ^2 '-c'
then pas ^1
    bind ^1.bin library -b ^1
else bind ^1.bin library -b ^1
endif
```

Example 2 might appear in a shell script. These lines compile the Pascal module named by the first argument ^1 if the second argument ^2 is `-c`. Then it binds the module with `library`. If the second argument is not `-c`, or if there is no second argument, the command simply binds the module.

SEE ALSO

More information is available. Type

help `abtsev` For information about abort-severity levels

NAME

import_passwd – create registry entries based on information in UNIX group and password files

SYNOPSIS

/etc/import_passwd [-i] [-a | -f] [-c] [-o org] -s pathname [-v]

DESCRIPTION

import_passwd is a mechanism for creating Apollo registry entries that are consistent with foreign password and group file entries. You should use **import_passwd** to ensure consistency between Apollo and foreign protection mechanisms when you

- Attach Apollo node(s) to an existing UNIX network
- Attach UNIX node(s) to an Apollo network
- Connect Apollo and UNIX networks

If the foreign password and group file entries do not exist in the Apollo registry, **import_passwd** will create them. If there are duplicate entries, **import_passwd** will follow your directions on how to handle them. (Note that reserved names and reserved UNIX IDs cannot be reassigned.)

The Process

The Apollo registry must exist before you can use **import_passwd**. If you are simply adding a few Apollo nodes to a foreign network, you can create a new, but empty, registry to meet this requirement. Once the registry exists, the registry server must be running, and you must be logged on as root.

As **import_passwd** processes, it

1. Examines the foreign group file and creates group entries in the registry.
2. Examines the foreign passwd file and creates person, organization, and account entries in the registry. The organization assigned is specified as input to **import_passwd**.
3. Reexamines the foreign group file and creates membership lists.

Conflicts

During this process, **import_passwd** may find conflicts in name strings (for example, in the foreign network, joe 102; in Apollo, joe 555) and in UNIX IDs (for example, in the foreign network, joe 102; in Apollo, ann 102). **import_passwd** provides a number of options to help resolve these conflicts.

The Favored Entry

The **-a** (favor Apollo entry) or **-f** (favor foreign entry) options specify which entry should be favored. A favored entry is retained as is. You are prompted to modify non-favored entries. (Note, however, that in some cases you may be prompted to modify a favored entry. For example, if the non-favored entry is a reserved name, you will be prompted to modify the favored entry.)

Name Conflicts

The `-i` option specifies that duplicate names are not in conflict but in fact, represent the same identity. Therefore, when duplicate names arise, no action is necessary. If you do not use the `-i` option, `import_passwd` resolves the name conflict by prompting for a name string for the non-favored entry.

UNIX ID Conflicts

The resolution of UNIX ID conflicts is also determined by the favored entry. If a conflict exists, you are prompted for a new UNIX ID for the non-favored entry.

Other Registry Entries

Except for names and UNIX IDs, all other information stored in the Apollo registry for an existing identity is retained.

New registry entries created by `import_passwd` are assigned the following values:

For Person and Group Entries:

`fullname =` " (empty)

`owner =` Same as the owner of the organization specified with the `-o` option. If no organization is specified, then the owner of the organization named "none".

`alias/primary =` Primary for first entry; alias for subsequent ones.

`projlist_ok =` Yes.

`passwd =` For groups only, taken from the group file.

`membership list =` For new groups only, all persons listed in the group file, and all persons with accounts in the password file with that group.

For Account Entries:

`abbreviation =` Shortest possible abbreviation that does not conflict with pre-existing Apollo accounts.

`account_valid =` True.

`gecos =` Same as UNIX password file.

`homedir =` Same as UNIX password file.

`shell =` Same as UNIX password file.

`passwd =` Same as UNIX password file. Note that you must modify or reset imported passwords before user authentication is possible and for the account to be usable in a pre-SR10 registry.

`passwd_dtm =` Date and time `import_passwd` was run.

`passwd_valid =` True.

OPTIONS

- i** Name strings are not in conflict, but represent the same identity.
- a (default)** Favor Apollo entries for conflicts.
- f** Favor foreign entries for conflicts.
- c** Run in check mode: Process the command, showing all conflicts, but make no requests for resolution.
- o *org*** *org* is the name of an Apollo organization to be assigned to all imported entries.
- s *pathname*** *pathname* is the path to the directory containing the foreign password and group files to be imported.
- v** Run in verbose mode: Generate a verbose transcript of. `import_passwd` activity.

NAME

inlib – install a user-supplied library

SYNOPSIS

inlib *pathname...*

DESCRIPTION

inlib installs a library at the current shell level; it remains installed until the shell that installed it exits. See the note below for information on loading a library that is used by all processes. The newly installed library will be used to resolve external references of programs (and libraries) loaded after its installation. (Thus, previously loaded libraries and programs will not be affected.)

Note that only those global references that are marked by the binder become visible, and that the default action of the binder is to leave globals unmarked. Therefore, you should take care to mark all appropriate globals when you bind your library.

inlib is an internal shell command.

NOTE

For performance reasons, we recommend that you use **bind -inlib** in place of **inlib**. **inlib** directs each subsequent invoked process to install the library; using **bind** with the **-inlib** option, the library is installed only with the required programs. See the **bind** command. You can create a library that is installed automatically in every process. This library resides in the file **/lib/userlib.private**. The procedure text in this library will be shared among all processes.

This library must be present at node start-up time in order to be installed. After copying your library to **/lib/userlib.private**, you must shut down the node and start it up again in order to use the library. Changes to the library also require rebooting the node to load the new routines.

Global names in **/lib/userlib.private** must not duplicate names used in Domain libraries.

pathname (required) Specify name of library file(s) to be installed. Multiple pathnames and wildcarding are permitted.

NAME

intm – install a type manager

SYNOPSIS

intm [*options*] *type_name* [*mgr_pathname*]

DESCRIPTION

intm installs a type manager for the *type_name*. The manager is copied into the type manager directory from *mgr_pathname*. If *mgr_pathname* is omitted, the file named *type_name* in the current directory is used. The **intm** command does not accept wild-cards.

type_name (required) Specify the type for which the manager is to be installed.

mgr_pathname (optional)

Specify the pathname of the manager object file to install for this type.

Default if omitted: object file is named *type_name*

OPTIONS

- n** *node_spec* Specify the node on which the type manager is to be installed. If this option is omitted, the type manager is installed on the current node.
- l** List the results of the operation.
- r** Replace an existing type manager if it exists.

EXAMPLES

```
$ intm example_type /mydir/my_example_mgr.bin
```

```
$ intm example_type /mydir/old_example_mgr.bin -n //remote_vol -l
"/mydir/old_example_mgr.bin" installed as the manager for
type example_type on volume //remote_vol.
```

SEE ALSO

More information is available. Type:

- help inty** For information on installing types
- help node_spec** For details about node specification syntax

NAME

inty – install a new type

SYNOPSIS

inty [*options*] *type_name* *source_volume* [*-n node_spec*]

DESCRIPTION

inty installs a type from one node to another. It installs both the type name and type manager on the target node (given by the *-n* option).

type_name (required) Specify the name of the type to be installed.

source_volume (required)

Specify the pathname of the source volume from which to copy the type name and type manager.

OPTIONS

- n node_spec* Specify the node on which the type is to be installed. You may also specify the entry directory of a volume mounted for software installation, as shown in the example below. If this option is omitted, the type is installed on the current node.
- l* List the results of the installation.
- r* Replace any existing type name/manager pair.

EXAMPLES

```
$ inty example_type //test_vol
Type "example_type" installed.
```

```
$ inty example_type //my_vol -n //test_vol -l
Type "example_type" installed on volume //test_vol.
```

SEE ALSO

More information is available. Type

- help intm** For information on installing type managers
- help node_spec** For details about node specification syntax

NAME

invol – initialize disk volumes

SYNOPSIS

/etc/invol (from shell)

ex invol (from mnemonic debugger)

DESCRIPTION

invol initializes physical disk volumes, creates logical volumes, and maintains badspot lists. Once initialized, a volume can be mounted with the `mtvol` command, or can be used to bootstrap the operating system, providing it contains the necessary files. `invol` prompts for all required information.

SUMMARY OF OPTIONS (Complete description follows.)

0	Exit.
1 [-fnb5uom]	Initialize virgin physical volume.
2 [-fnb5u]	Add a logical volume.
3 [-fnb5]	Re-initialize an existing logical volume.
4	Delete a logical volume.
5	List logical volumes.
6	List badspots on disk or volume.
7	Create physical badspot list.
8	Create or modify a Domain/OS paging file.
9	Add to existing badspot list.
10	Display/change sector interleave factor.
11	Remove from existing badspot list.

FLAGS SUMMARY

-u	Use defaults
-f	Don't re-format disk
-o	Pre-SR10 format
-n	Make non-bootable volume
-b	Apply BSD UNIX ACLs
-5	Apply SysV UNIX ACLs
-m	Make a multi-disk set

FULL DESCRIPTIONS OF OPERATIONS

0 Exit

1 [-fmb5uom]

Initialize a virgin physical volume.

Every new disk must be initialized before it can be used. When you initialize a new disk, all existing data on the disk are overwritten. Do not initialize a disk that contains any data you need to save. We initialize Winchester disks during the manufacturing process, before installing the system software.

To initialize a new disk, follow this procedure:

- A. invol asks which option to perform. Type 7 to create or replace the badspot list. (See "Recording Badspot Information"). Type 9 if you want to add to the existing badspot list. Otherwise, type 1 to initialize a new physical volume.
- B. Specify the type of disk to initialize. invol prompts with:

```
Select disk: [w=Winch|s=Storage mod|f=Floppy|q=Quit]
             [ctrl#:] [unit#]
```

Typing q (as always) will exit the program.

invol (as well as salvol and fixvol) can deal with multiple controllers of the same type. The encoding of the controller# and unit# is as follows:

```
w           Winchester ... Controller #0 ... Unit #0
w1          Winchester ... Controller #0 ... Unit #1
w1:         Winchester ... Controller #1 ... Unit #0
w1:2       Winchester ... Controller #1 ... Unit #2
```

Thus a single character N following the slw specifies a unit# on the first (0'th) controller. If this character is followed by a ':' character, it is taken to be a controller specifier which is then followed by an optional unit specifier.

- C. invol asks for the name of the physical volume.
- D. Choose one of the following verification options:
 - 1 - No verification
 - 2 - Write all blocks on the volume
 - 3 - Write and reread all blocks on the volume

If you choose no verification (option 1), invol does not read or write to the disk, except to create the volume structure. This option is the fastest, but means that the disk is not verified until it is mounted and read or written.

If you choose the second option, invol attempts to write to each block on the disk. The third option, writing and rereading all blocks on the volume, is the safest but also the slowest. For example, to format a complete 33MB Winchester, option 1 requires about five minutes, option 2 requires about fifteen minutes, and option 3 requires about 30 minutes.

If a floppy disk is initialized with invol on a busy node, there is a small chance that a format operation will fail, but that the failure will not be reported to invol. For this reason, invol writes each block during floppy initialization, even for verification option 1. If a write fails after an apparently successful format, invol will print the message:

```
format failed for daddr <disk_address>:<write status>
-- retrying format
```

and will reformat (and rewrite) the track in error. This happens whether or not the floppy has been previously initialized.

- E. Enter the average file size, when prompted:

```
Expected average file size, in blocks
(CR for default-5 blocks):
```

Press <RETURN> to accept the default value of 5 blocks. Supplying a relatively accurate value for the average file size can save space on the disk, because the volume table of contents (a system table) will be allocated more efficiently.

- F. invol requests the size (in blocks) and name of each logical volume to be created. After each entry, invol tells you how much space remains. After entering the size and name of all logical volumes, enter a blank line to terminate input. A physical volume can contain up to 10 logical volumes. For example:

```
There are 1231 blocks available.

volume 1: 1231,vol1
```

The logical volume size must be at least 30 blocks, and must be a multiple of the track size for the disk. If you specify a logical volume size

that is not a multiple of the track size, invol rounds it up to the next multiple track size, and informs you. Note that the physical volume label occupies the first block on the volume. Thus, the size of the first logical volume is always one less than a multiple of the track size.

Logical volume names are optional, and are used only when invol lists the logical volumes on the disk (option 5). You cannot change the name of a logical volume after creating it.

- G. invol requests badspot information by asking whether or not you wish to use the prerecorded badspot list shipped with the disk. Answer y[es]. To erase the existing list, answer no. If you want to initialize the physical badspot list on a virgin disk, use option seven, not option one. Use option nine to add to an existing list. You must have a hardcopy of the badspots in order to enter them. invol has retained the badspot prompt in option one only for compatibility with existing shell files. After your affirmative response, invol displays the badspot list, indicating the physical disk address, cylinder, head, sector, and byte offset range.

If, in later operations, you wish to provide your own badspot information, these can be entered in one of several formats:

If the disk is a floppy: only HEX disk (block) addresses may be entered.

If a multi-disk set was assigned, only HEX physical disk (block) addresses can be entered (as reported by salvol, lsyserr or disk_err). These disk addresses are relative to the entire set. For example: for a multi-disk set, daddr 0 is on the 1st drive, 1 is on the 2nd one, etc.

Otherwise, the user has the option of entering the following:

HEX physical disk (block) addresses
 DECIMAL cylinder-head byte_offset1 [byte_offset2 ...]

up to 8 byte offsets can be
 specified per-line (for the
 same cylinder/head)

HEX \$cylinder-head sector1 [sector2 ...]

up to 8 sectors can be specified
 specified per-line (for the
 same cylinder/head)

Input continues until the user enters a blank line at which time he is given an opportunity to start over again (ignoring everything entered thus far).

If the disk contains logical volume badspots lists (see earlier) the badspot changes are propagated to these lists as well.

- H. **invol** initializes the disk. As formatting proceeds, **invol** displays milestone messages to report its progress. It also displays a message for each volume it initializes, and another when it completes.
- I. **invol** asks if you have any more requests. Type **y[es]** to return to step A, or **n[o]** to return to the calling program (shell or Mnemonic Debugger).

2 [-fmb5u]

Using option 2, you can partially initialize a volume, that is, add logical volumes to a physical volume, while preserving the existing logical volumes. Follow this procedure:

- A. **invol** asks which option to perform. Type **2** to partially initialize a disk.
- B. Specify the type of disk to initialize. (See option 1 step B.)
- C. **invol** prints a list of the logical volumes and vacancies on the disk. If the disk has more than one vacancy, **invol** asks where to place the new logical volume by requesting a vacancy number. Indicate the vacancy that you want **invol** to use by entering its number. If there are logical volumes following the vacancy that you choose, **invol** prints a warning message and then automatically increments the volume numbers of those succeeding volumes by one.
- D. Choose a verification option for the logical volume being initialized. (See option 1 step D.)
- E. Enter the expected average file size, in blocks. (See option 1 step E.) Press **<RETURN>** for the default value, 5 blocks.
- F. Enter the name and size of each logical volume to be formatted. (See option 1 step F.) After each specification, **invol** informs you of how much space is available. Terminate input with a blank line. A physical volume may have up to ten logical volumes.
- G. Enter badspot information. (See option 1 step G.) Terminate badspot entry with a blank line.
- H. Enter the name of the physical volume. (See option 1 step C.)
- I. **invol** asks if you have any more requests. Type **y[es]** to return to step A, or **n[o]** to return to the calling program (shell or Mnemonic Debugger).

3 [-fmb5]

You can reinitialize a logical volume, retaining its size and name, with option 3. All existing data in the volume will be lost. This option is useful for reinitializing floppy disks, where one logical volume typically occupies the entire physical volume.

To reinitialize a single logical volume, use the following procedure:

- A. `invol` asks which option to perform. Type 3 to reinitialize a logical volume.
- B. Specify the type of disk to initialize. (See option 1 step B.)
- C. `invol` prompts for the # (1..N) of the logical volume to be re-initialized.
- D. Choose a verification option: no verification, write all blocks, or write and reread all blocks. (See option 1 step D.)
- E. Enter the expected average file size, in blocks. (See option 1 step E.) Press <RETURN> for the default value, 5 blocks.
- F. `invol` asks if you have any more requests. Type y[es] to return to step A, or n[o] to return to the calling program (shell or Mnemonic Debugger).

4 Delete a logical volume.

To delete a logical volume, use the following procedure:

- A. `invol` asks which option to perform. Type 4 to delete a logical volume.
- B. Specify the type of disk from which the volume will be deleted. (See option 1 step B.)
- C. Enter the number of the logical volume to delete. You can determine the logical volume numbers present on a disk with option 5.
- D. `invol` asks if you have any more requests. Type y[es] to return to step A, or n[o] to return to the calling program (shell or Mnemonic Debugger).

Note: `invol` renumbers the logical volumes following the deleted volume.

5 Listing logical volumes

Logical volumes on the disk are displayed. Pre-SR10 format logical volumes are flagged as such in the output. To list the logical volumes on a disk, use the following procedure:

- A. `invol` asks which option to perform. Type 5 to list the logical volumes on a disk.

- B. Specify the type of disk. (See option 1 step B.) invol lists the volumes on that disk.
- C. invol asks if you have any more requests. Type y[es] to return to step A, or n[o] to return to the calling program (shell or Mnemonic Debugger).

6 List badspots on disk or volume.

To list the badspots in one or more logical volumes, or for the physical volume, use the following procedure:

- A. invol asks which option to perform. Type 6 to list badspots.
- B. Specify the type of disk. (See option 1 step B.)
- C. Specify the badspots to be listed, by entering one of the following:

m[fg] For ESDI drives only:

The contents of the manufacturer supplied badspot list is displayed. Physical disk (block) addresses are displayed in HEX along with the corresponding HEX (as opposed to decimal pre-SR10) cylinder/head/sector addresses and DECIMAL byte offset range.

Note: Users should generally have no reason to use this option as this list is usually copied to the physical badspot list/cylinder as soon as a disk is received. On some disks, moreover, this manufacturer's list is destroyed as soon as the disk is invol'd (option 1).

p[hys] For Winchester and Storage-Module drives only:

Displays the contents of the physical badspot list. Physical disk (block) addresses are displayed IN HEX along with the corresponding HEX (as opposed to decimal pre-SR10) cylinder/head/sector addresses and DECIMAL byte offset range.

If the specified disk is the primary drive for a multi-disk set, the physical disk addresses are relative to the entire set and a disk identifier is displayed along with the cylinder/head/sector. This disk identifier consists of a controller number and drive/unit number. For example:

```
phys cylinder head sector byte offset C_num Drv_Unit
daddr                range
...
...
123a9f 12d e 5 7123-8034 0 2
```

If the specified disk is a non-primary member of a multi-disk set (i.e., the `s` option was used to examine a single drive only, see the `-m` option below), then the physical disk addresses that are displayed are relative to that drive and do not correspond in any way to logical/physical disk addresses for the disk-set as a whole (i.e., as displayed by `lyserr` or `disk_err`).

- n* Badspots that lie within the specified logical volume are displayed: *n* can be any integer from 1 through 10. Both logical and physical disk addresses are displayed in HEX.

The specified disk must hold a valid `pv` and `lv` labels. The primary member of a multi-disk set must have been specified.

Note: cylinder/head/sector values are not displayed.

For a multi-disk set, the primary drive of the set must have been specified earlier. Physical disk addresses are relative to the entire set.

- a[III] Badspots for all logical volumes on the disk (or multi-disk set) are displayed: this format is identical with that of 1..10 discussed above.

- D. `invol` asks if you have any more requests. Type `y[es]` to return to step A, or `n[o]` to return to the calling program (shell or Mnemonic Debugger).

7 Create physical badspot list.

Note: This option acts upon a single disk drive, regardless of whether it is a member of a multi-disk set or not. Generally speaking, this operation is run on a disk as it arrives from the factory and should not need to be performed again.

Using option 7, you can create or replace the badspot list on the disk. (Use option 9 to add badspots to an existing badspot list.)

- A. `invol` asks which option to perform. Type 7 to enter the disk's badspot list.
- B. Enter the location of the badspots, one per line. (See option 1 step G for the proper format.) Terminate badspot information with a blank line.
- C. After you have typed in the list, `invol` asks you to check for errors. If you made any errors in the list, you must retype the entire list by returning to step A and beginning again.
- D. `invol` asks if you have any more requests. Type `y[es]` to return to step A, or `n[o]` to return to the calling program (shell or Mnemonic Debugger).

8 Create or modify a Domain/OS paging file on an existing logical volume.

You can create an operating system file or modify the size of an existing one. The Domain/OS paging file is required if you intend to run the operating system off of this logical volume.

To create or modify a Domain/OS paging file, perform the following steps:

- A. `invol` asks which option to perform. Type **8**.
- B. Specify disk type. (See option 1 step B.)
- C. Specify logical volume number. The logical volumes present on a disk may be listed using option 5.
- D. If a Domain/OS paging file already exists on this volume, `invol` displays the file's current size and asks if you want to change it. If you reply `y[es]`, `invol` proceeds to step E. If you reply `n[o]`, `invol` skips to step F.
- E. `invol` prompts you to enter the number of pages you want in the OS paging file. Press `<RETURN>` to use the default, 352 pages. Type **0** (zero) to delete an existing paging file, or specify any number of pages between 1 and 288. If the size you enter is larger than the current Domain/OS paging file, `invol` displays milestones as it initializes new disk records.
- F. `invol` asks if you have any more requests. Type `y[es]` to return to step A, or type `n[o]` to return to the calling program (shell or Mnemonic Debugger).

9 Add to existing badspot list.

Using option 9, you can add to the disk's existing badspot list. Run `salvol` when this operation is done.

- A. `invol` asks which option to perform. Type **9** to add to the disk's badspot list.
- B. Enter the location of the badspots, one per line. (See option 1 step G for the proper format.) Terminate badspot information with a blank line.
- C. After you have typed in the list, `invol` asks you to check for errors. If you made any errors in the list, you must retype the entire list by returning to step A and beginning again.
- D. `invol` asks if you have any more requests. Type `y[es]` to return to step A, or `n[o]` to return to the calling program (shell or Mnemonic Debugger).

10 Display/change sector interleave factor for a logical volume.

Using option 10, you can set or display the sector interleave factor for a volume. The correct interleave factor is set when a logical volume is created. However, as performance improvements are made, it may become necessary to change it to achieve optimal block read/write rates. Operation 10 displays the current value and the optimal value which we recommend.

- A. invol asks which option to perform. Type 10 to set or display the sector interleave factor.
- B. Specify disk type. (See option 1 step B.)
- C. invol displays a list of logical volumes for that physical volume. Specify the appropriate logical volume number. invol then displays the current sector interleave factor and the value which we recommend.
- D. invol prompts for the new interleave factor. If you do not wish to change the interleave factor, enter a carriage return.
- E. invol asks if you have any more requests. Type y[es] to return to step A, or type n[o] to return to the calling program (shell or Mnemonic Debugger).

11 Remove badspots from existing badspot list.

Using option 11, you can subtract from the disk's existing badspot list. Run salvol when this operation is done.

- A. invol asks which option to perform. Type 11 to remove from the disk's badspot list.
- B. Enter the location of the badspots, one per line. (See option 1 step G for the proper format.) Terminate badspot information with a blank line.
- C. After you have typed in the list, invol asks you to check for errors. If you made any errors in the list, you must retype the entire list by returning to step A and beginning again.
- D. invol asks if you have any more requests. Type y[es] to return to step A, or n[o] to return to the calling program (shell or Mnemonic Debugger).

FULL DESCRIPTION OF FLAGS

- u** Use defaults. You are prompted for as little information as possible. A physical volume is automatically chosen as follows (for option 1):

```
pv_wxy_year.month.day    w: disk type (or s or f)
                          x: controller #
                          y: unit #
                          year.month.day: today's date
```

A single logical volume spanning the entire physical volume (for options 1 and 2) is constructed and chosen as follows:

```
lv_year.month.day
```

- n** A non-bootable volume is constructed. By default, invol places the following objects on each logical volume that it constructs:

```
lost+found.list          lost+found file for Salvol's later
                          use space is reserved for sysboot
* sysboot                (although the cpboot command must
                          be issued to actually install it
                          there)
                          //          local network root
                          /           entry directory for volume
* /sys
* /sys/node_data
* /sys/node_data/system_logs
```

If the **-n** flag is specified, the objects marked ***** are not placed on the disk. Although the disk can be mounted, it can not be used as a boot volume (unless it is re-invol'd).

- f** Do not physically reformat the disk. By default, invol performs the very time-consuming task of re-formatting every track on the disk. This flag bypasses that operation and causes invol to execute very quickly, especially so if it is coupled with Verification option 1. Normally, it is only necessary to reformat disks as they arrive from the factory or after you have reason to suspect that the physical formatting is damaged or otherwise corrupted.

- b** BSD style initial acls (for inheritance) are placed onto /sys and /sys/node_data when they are constructed:

```
owner/org:      ids from creating process
group:         id taken from parent directory
               (set to wheel for /sys and
               /sys/node_data)
<all> rights:  specified (usually from umask)
               at create time
```

- 5** SysV style initial acls (for inheritance) are placed onto /sys and /sys/node_data when they are constructed:

```
owner/group/org:  ids from creating process
                  rights specified (usually
                  from umask) at create time
world rights:    "      "      "
```

By default, Aegis inclusive initial acls are applied.

```
owner/group/org:  ids from creating process
                  rights set to [ignore]
world rights:    pwx (wide-open)
```

- o** Use only with option 1. A pre-SR10 format disk is configured which can be mounted under a pre-SR10 version of the operating system. By default, a SR10 format disk is created which cannot be mounted by a pre-SR10 node. The presence or absence of this flag controls whether subsequent logical volumes on this volume will have SR10 format (the new file system, acls and directories) or the pre-SR10 one.
- m** Use after option 1 to group multiple physical disks together into a multi-disk set which thereafter will appear to be a single large disk (to salvol, mtvol, dmtvol, etc.).

If you answer the invol prompt with **l -m** to indicate that a multi-disk group is to be configured, you will get the following prompts:

- A. Select disk: [w=Winchls=Storage modf=Floppy/q=Quit][ctrl#:][unit#]
 invol prompts for the identity of the first of the disks. Assume that the user responds with **w0:1**. Subsequently, this drive becomes the primary disk of the set: it is the one which must be specified in order to mount or otherwise use the set.

- B. How many disks will you be grouping together?
 invol prompts for the number of disks to be collected into the set.
 Currently, the only legal responses are 1, 2 or 4.
- C. Enter the striping option.
 invol prompts for the algorithm to be used in spreading disk blocks across the multiple drives in the set:
1. Sector striping (subsequent sectors to different drives)
 This is usually the choice for DN10000 workstations. Subsequent disk blocks are sent to alternate disk drives (e.g., disk block N is sent to a different drive from N-1 and N+1); this is the "classical" definition of disk striping.
 2. Cylinder striping (subsequent cylinders to different drives)
 This is usually the choice for 68000-based workstations. The cylinders from the various disks are "stacked" on top of one another so that it appears that each cylinder has N (2 or 4) times as many platters. Subsequent disk blocks will reside on the same drive unless a cylinder boundary is crossed, in which case we drop to the next disk in the set or wrap to the next cylinder of the first (primary) disk drive.
- D. Physical volume name:
 invol prompts for the name of the physical volume.
 Enter remaining members of disk group:
 Select disk: [w=Winchls=Storage modlf=Floppy]q=Quit][ctrl#:][unit#]
 Select disk: [w=Winchls=Storage modlf=Floppy]q=Quit][ctrl#:][unit#]
 invol prompts for the identity of the additional disk drives in the set. Obviously, no two drives in the set can have identical controller and unit numbers.
 This **-m** flag for option 1 is allowable only if:
- A. invol is running under an SR10 operating system.
 - B. An SR10 format disk is being configured.
- C. A Winchester drive is selected. Only Winchester drives may be configured as a multi-disk set in this fashion. All drives in the set must have identical geometries: specifically, the `blocks_per_track`, `sects_per_track`, `tracks_per_cylinder`, `blocks_per_volume`, `blocks_per_physical_volume`, `physical_badspot_daddr` and `physical_diagnostic_daddr` attributes must all agree.
- Note: Be aware that what is constructed is a single physical volume spanning multiple disk drives. Any logical volumes later built span all disk drives of the set. There is no way to cause a single file or logical volume to be limited to a single drive of the set. Contained within the `pv` label of the primary drive (as

well as the others) is information identifying the other drives of the set as well as consistency data to detect the re-invol or replacement of any of the drives.

Relation of invol options to multi-disk sets:

The following invol options, when given the specifier of the primary drive of a set, will operate upon the entire set:

- 2 add a logical volume to an existing physical volume.
- 3 re-initialize an existing logical volume.
- 4 delete a logical volume.
- 5 list logical volumes.
- 6 list badspots: unless [s] is specified – <see below>.
- 8 create Domain/OS paging file.
- 9 add to badspot list: unless [s] is specified – <see below>.
- 10 changing interleave factor for a logical volume.
- 11 delete from badspot list: unless [s] is specified – <see below>.

Several invol options will only operate upon a single disk drive – regardless of whether it is a member of a multi-disk group:

- 1 as detailed above, the specified drive is re-initialized regardless of its current status as a member of some multi-disk set ... optionally, a new multi-disk set can be established.
- 7 a physical badspot list on the specified drive is constructed (this option needs to be run when the drive arrives from the factory, and not again).

Several invol options allow the user to specify that only a single drive is to be assigned/processed, even if that drive is a member of a multi-disk set:

- 6 list badspots : only the physical (or manufacturer's) badspot list can be shown in this case.
- 9 add to badspot list : badspots may be entered as either physical disk addresses or as cylinder-head-sector triplets – both relative to that single drive as if it was not a member of a multi-disk set.
- 11 delete from badspot list : see the above discussion for 9.

For these cases, the "assign" prompt given to the user specifies that the user is allowed to follow the disk specifier with a "s" (<space>, "s") to indicate that even though the disk is a member of a multi-disk set, only the specified (single) disk is to be accessed. For example:

```
Select disk: [w=Winch|s=Storage mod|f=Floppy|q=Quit]
             [ctrl#:] [unit#]   w0:1 s
```

NAME

ios_test – test ios_\$ calls

SYNOPSIS

ios_test [-init]

DESCRIPTION

ios_test is a program for testing type managers that manage input and output to objects. **ios_test** allows you to open a stream to any type of object and then use selected IOS calls on the open stream. With **ios_test**, you can open streams to existing or new objects. For more information on using **ios_test** to test type managers, see *Using the Open System Toolkit to Extend the Streams Facility*.

OPTIONS

-init Call the **ios_\$initialize** routine (within a type manager) at start-up time.

COMMANDS SUMMARY

ios_test prompts for commands. Any valid, unambiguous prefix of one of the following commands will suffice. Each command calls the IOS call with a similar name. For example, the **close** command calls **ios_\$close**.

Syntax	Function
change_path_name <i>stream-id</i> <i>pathname</i>	Changes the pathname of an object.
close <i>stream-id</i>	Closes a stream.
create <i>create-mode</i> [<i>open-options</i>] <i>pathname</i> <i>typename</i>	Creates an object and opens a stream to it.
delete <i>stream-id</i>	Deletes an object and closes the associated stream.
dup <i>stream-id</i> <i>stream-id</i>	Creates a copy of a specified stream ID.
equal <i>stream-id-1</i> <i>stream-id-2</i>	Determines whether two stream IDs refer to the same object.
export <i>stream-id</i>	Simulates stream passing via a pgm_\$invoke system call. This command tests a type manager's export and import procedures.
force_write <i>stream-id</i>	Forcibly writes an object and the directory containing the object to stable storage.
get [<i>put-get-option</i>] <i>stream-id</i> <i>count</i>	Copies data from a stream into a buffer.
inq_byte_pos [<i>pos-opt</i>] <i>stream-id</i>	Returns the byte position of the stream marker. If you omit a position option, the default is the current position of the stream marker.
inq_cur_rec_len <i>stream-id</i>	Returns the length of the record at the current stream marker.

inq_file_attr <i>stream-id</i>	Returns object usage attributes.
inq_flags <i>stream-id</i>	Returns the attribute set of an object's type manager.
inq_full_key [<i>pos-opt</i>] <i>stream-id</i>	Returns a full seek key. If you omit a position option, the default is the current position of the stream marker.
inq_path_name [<i>name-type</i>] <i>stream-id</i>	Returns the pathname of the object to which a stream is open. If you omit a <i>name-type</i> , the default is <code>-root</code> .
inq_rec_pos [<i>pos-opt</i>] <i>stream-id</i>	Returns the record position of the stream marker. If you omit a position option, the default is the current position of the stream marker.
inq_rec_rem <i>stream-id</i>	Returns the number of bytes remaining in the current record.
inq_rec_type <i>stream-id</i>	Returns the record type of an object.
inq_short_key [<i>pos-opt</i>] <i>stream-id</i>	Returns a short seek key. If you omit a position option, the default is the current position of the stream marker.
inq_type_uid <i>stream-id</i>	Returns the type UID of an object.
locate <i>stream-id count</i>	Reads data from a stream, returning a pointer to the data (rather than copying the data to a buffer).
open [<i>open-options</i>] <i>pathname</i>	Opens a stream to an existing object.
put [<i>put-get-options</i>] [-nl] <i>stream-id string</i>	Writes data into an object. The -nl option inserts a newline character at the end of the string, the default writes only the data.
readdir <i>stream-id maxcnt bufsize</i>	Reads up to <i>maxcnt</i> dir entries or so long as there is space in <i>bufsize</i> (or EOF is reached).
replicate <i>stream-id stream-id</i>	Creates a copy of a specified stream ID.
rewinddir <i>stream-id</i>	Rewinds dir <i>stream-id</i> to beginning-of-file (BOF).
seek [-relative [-minus]] [-record] <i>stream-id count</i>	Performs an absolute or relative seek using byte or record positioning. If you omit the -relative option, the default is an absolute seek. If you omit the -minus option, the default is to seek forward, towards the end of the file. If you omit the -record option, the default is to seek by bytes.

<code>seekdir stream-id key</code>	Positions to key within dir <i>stream-id</i> (key must be returned by <i>telldir</i>)
<code>seek_full stream-id recadr byteadr</code>	Performs a seek using a full (8-byte) seek key.
<code>seek_short stream-id key</code>	Performs a seek using a short 4-byte seek key.
<code>seek_to_bof stream-id</code>	Positions the stream marker to the beginning of an object.
<code>seek_to_eof stream-id</code>	Positions the stream marker to the end of an object.
<code>switch stream-id stream-id</code>	Switches a stream from one stream ID to another stream ID.
<code>telldir stream-id</code>	Returns a key that refers to current position in dir <i>stream-id</i>
<code>truncate stream-id</code>	Deletes the contents of an object following the current stream marker.

Use one of the following to specify *create-mode*. These options correspond to the `ios_$create_mode_t` data type.

<code>-no_pre_exist</code>	Returns an error if object already exists.
<code>-preserve</code>	Saves contents of object, if it exists, opens object, and positions stream marker at BOF.
<code>-recreate</code>	Deletes existing object and creates new one of the same name.
<code>-truncate</code>	Opens object, then truncates the contents.
<code>-make_backup</code>	Creates a backup (.bak) file when closed.
<code>-loc_name_only</code>	Creates a temporary unnamed object, uses pathname to specify location of object, and locates it on the same volume.

Use one of the following to specify *name-type*. These options correspond to the `ios_$name_type_t` data type.

<code>-leaf</code>	Specifies leaf name regardless of object's name.
<code>-ndir</code>	Specifies leaf name if object's name is a name in current naming directory; otherwise, specifies full pathname.

-node	Specifies name relative to the root directory if object is a name in boot volume; otherwise, specifies full pathname.
-node_data	Specifies leaf name if object's name is a name in current <code>node_data</code> directory; otherwise, specifies full pathname.
-root	Specifies full pathname, for example, <code>//node/sid/file</code> .
-wdir	Specifies leaf name if object's name is a name in current working directory; otherwise, specifies full pathname.

Use one or more of the following to specify *open-options*. These options correspond to the `ios_$open_options_t` data type.

-no_d[elay]	<code>ios_\$open</code> does not wait for the open to complete before returning.
-w[rite]	Permits writing data to a new object.
-unreg[ulated]	Permits concurrent writing (unregulated read and write access) to the object.
-end_of_file	Positions stream marker at EOF at open.
-inq[uire_only]	Opens object for attribute inquiries only.

Use one of the following to specify *pos-opt*. These options correspond to the `ios_$pos_opt_t` data type.

-bof	Returns key for EOF marker.
-eof	Returns key for BOF marker.

Use one or more of the following to specify *put-get-options*. These options correspond to the `ios_$put_get_opts_t` data type.

-cond	Gets or puts data conditionally. If the data is not available, returns with a status indicating that condition.
-pre[view]	Determines if a put/get would succeed, but does not actually perform data transfer.

-part[ial_record]	Puts the data, but does not terminate the record.
-no_rec_[bndry]	Ignore record (line) boundaries.

DEBUGGING MANAGERS

Under normal conditions, user-written managers are dynamically loaded into the opener's address space. While you can use `ios_test` to test such managers, the manager code itself can not be debugged using `debug` at the present time.

To debug managers using `ios_test`, you must follow the convention that your manager contains no "main program" (PROGRAM in Pascal, "main" in C). Instead, the initialization for your manager (the part that calls `trait_$mgr_dcl`, etc.) should be placed in a procedure named "`ios_$initialize`". To debug your manager module using `ios_test`, bind all the pieces of your manager together with `/com/ios_test`. Then use `debug` on the result of the bind and give the `-init` option. For example

```
$ bind -b my_itest <<!
my_mgr.bin
my_mgr_uid.bin
/sys/traits/io_traits
/com/itest
!
$ debug -src my_itest -init
```

NAME

iso – convert files to ISO format

SYNOPSIS

```
french_to_iso  input_file output_file  
german_to_iso input_file output_file  
nor.dan_to_iso input_file output_file  
swedish_to_iso input_file output_file  
swiss_to_iso  input_file output_file  
uk_to_iso     input_file output_file
```

DESCRIPTION

These utilities convert files written with the overloaded 7-bit national fonts to the International Standards Organization (ISO) 8-bit format. The overloaded fonts include any with a specific language suffix (for example, *f7x13.french*, or *din_f7x11.german*). If you created and/or edited a file using one of the national fonts, that file is a candidate for conversion.

You are not required to convert files, but probably will want to because

1. The overloaded fonts replace certain ASCII characters with national ones, have that subset of ASCII characters and the national characters in one file. The 8-bit fonts available as of SR10 include all the ASCII characters and the national characters.
2. The 8-bit fonts also include a wider range of national characters, so you can enter any character in any western European language. This was not always possible with the overloaded fonts. For example, there was not enough space in the overloaded font to include all the French characters, but they all exist in the 8-bit fonts.

There are two parameters to the conversion utilities. You must provide the name of the file you want to convert (*input_file*) and your *output_file*. If *output_file* already exists, the utilities abort.

The default location for the utilities is */usr/apollo/bin*.

ISO

Aegis

ISO

FILES

<code>/usr/apollo/bin/french_to_iso</code>	Converts overloaded French to ISO format
<code>/usr/apollo/bin/german_to_iso</code>	Converts overloaded German to ISO format
<code>/usr/apollo/bin/nor.dan_to_iso</code>	Converts overloaded Norwegian/Danish to ISO format
<code>/usr/apollo/bin/swedish_to_iso</code>	Converts overloaded Swedish/Finnish to ISO format
<code>/usr/apollo/bin/swiss_to_iso</code>	Converts overloaded Swiss to ISO format
<code>/usr/apollo/bin/uk_to_iso</code>	Converts overloaded U.K. English to ISO format

DIAGNOSTICS

All messages are generally self-explanatory.

EXAMPLE

§ `french_to_iso dictionnaire new.dictionnaire`

NAME

kbm -- set/display keyboard characteristics

SYNOPSIS

kbm [-c *args*] [-l *args*] [-s *args*]

DESCRIPTION

kbm allows you to set the characteristics for the keyboard. Settable characteristics are the compose key(s), and the long and short shift key(s) on the Domain multinational keyboard. The compose key is used to compose characters of the latin-1 character set that do not have corresponding keys on the keyboard. Long and short shift are used to toggle the alternate key labels on the multinational keyboards.

OPTIONS

If no options are specified, **kbm** displays the current keyboard type and characteristics.

- c *args* Set compose keys to those specified by list *args*.
- l *args* Set long shift keys to those specified by list *args*.
- s *args* Set short shift keys to those specified by list *args*.

A key list is a list of function key names separated by commas. The following keys are allowable:

Key Name	Positions
l1-lf	Left function keys
r1-r6	Right function keys
f0-f9	Center function keys
np0-np9, npa-npg, npp	Numeric pad
tab,	TAB
bs	BACKSPACE
ar, al	ALT keys (multinational keyboard only)

Shifted keys are specified by appending an "s" to the key name, control keys by appending a "c", the up transition by appending an "u"; for example ar, ars, arc, aru .

To disable a function specify a key name of "none".

EXAMPLES

Display current characteristics

```
$ kbm
keyboard: 3
compose:  f5
long_alt:  als,ars
short_alt: al,ar
```

Set long shift keys to shifted ar and shifted al; short shift keys to al and ar.

```
$ kbm -l als,ars -s al,ar
```

Disable the compose function.

```
$ kbm -c none
```

NAME

lamf – laminate files

SYNOPSIS

lamf [*pathname...*] [*-s string*]

DESCRIPTION

lamf laminates the named files to standard output. That is, it concatenates the first lines of all input files, sequentially, and writes the result to standard output; and so on for the next input lines. If the files contain different numbers of lines, null lines are used for the missing lines in the shorter files.

NOTE

To insert a newline character between lines, use the escape sequence, **@n**, as a string argument. (See Example 4, below.)

ARGUMENTS

pathname (optional) Specify name(s) of file(s) to be laminated to standard output. Multiple pathnames are permitted, separated by blanks. The default is to read standard input for input lines. Use a hyphen (**-**) to specify standard input in a list of filenames.

OPTIONS

-s string Specify a string of text to be placed in each output line at the point it appears in the command argument list. *string* may not exceed 300 characters. Strings containing embedded spaces must be in quotation marks (" ").

EXAMPLES

1. Laminate files **mary** and **fred** and write results to standard output.

```
$ lamf mary fred
```

2. Laminate lines from **jan**, standard input, and **george**, in that order.

```
$ lamf jan - george
```

3. **\$ lamf -s "A line from A: " a -s ", and from B: " b**

produces:

A line from A: first line from a, and from B: first line from b

Note that the text strings inserted are not bound in any way to the position of the pathname arguments: you may place strings wherever you please. Those strings that contain literal blanks must be enclosed in quotation marks, as above.

4. Escape sequences are valid in string arguments. For example

\$ lamf mary -s @n fred

Insert a newline character between each line from **mary** and **fred**, thus interleaving the lines from the two files.

NAME

las – list objects mapped into the address space

SYNOPSIS

las [*options*]

DESCRIPTION

las produces a list of objects mapped into the address space. Information printed includes the virtual address range, the starting address within the object, and its pathname if available (in that order).

This command is most useful for system-level debugging.

OPTIONS

If no options are specified, **las** lists the address space of the current process.

- all** List all address space, including that occupied by Aegis.
- f[rom] address** Begin listing at the hexadecimal *address* specified.
- t[o] address** End listing at the hexadecimal *address* specified.

EXAMPLES

1.

```
$ las
```

VA Range	Obj Start	Pathname
8000 - 17FFF	0	/sys/node_data/global_data
18000 - 2FFFF	0	/lib/pmlib
30000 - 37FFF	0	/lib/syslib.peb
38000 - 4FFFF	0	/lib/kslib
50000 - 57FFF	0	/lib/trait_type_lib
58000 - 67FFF	10000	/sys/node_data/global_data
68000 - 9FFFF	0	/lib/streams
A0000 - A7FFF	0	/lib/vfmt_streams
A8000 - BFFFF	0	/lib/error
C0000 - E7FFF	0	/lib/swtlib
E8000 - F7FFF	0	/lib/ftnlib
F8000 - FFFFF	0	/lib/pbulib
100000 - 127FFF	0	/lib/gprlib
128000 - 14FFFF	0	/lib/clib
150000 - 157FFF	0	/lib/lisp_initlib
158000 - 15FFFF	0	/sys/node_data/global_rws
160000 - 16FFFF	20000	/sys/node_data/global_data
170000 - 187FFF	0	/lib/shlib
188000 - 19FFFF	0	/lib/tfp


```

1A0000 - 1BFFFF      0 /lib/dialoglib
1C0000 - 1C7FFF      0 /sys/node_data/ipc_data
1D0000 - 1D7FFF    30000 /sys/node_data/global_data
200000 - 2AFFFF      0 -- temporary file --
2B0000 - 2B7FFF      0 /sys/node_data/dm_mbx
2B8000 - 2BFFFF      0 /com/sh
2C0000 - 2C7FFF      0 -- temporary file --
2C8000 - 2CFFFF      0 /com/las
2D0000 - 2F7FFF    B0000 -- temporary file --
BC0000 - BCFFFF      0 /help_area/worksite
BD0000 - BDFFFF      0 /jttj

```

2944 KB mapped.

2.

\$ las -from 188000

VA Range	Obj Start	Pathname
188000 - 19FFFF	0	/lib/TFP
1A0000 - 1BFFFF	0	/lib/dialoglib
1C0000 - 1C7FFF	0	/sys/node_data/ipc_data
1D0000 - 1D7FFF	30000	/sys/node_data/global_data
200000 - 2AFFFF	0	-- temporary file --
2B0000 - 2B7FFF	0	/sys/node_data/dm_mbx
2B8000 - 2BFFFF	0	/com/sh
2C0000 - 2C7FFF	0	-- temporary file --
2C8000 - 2CFFFF	0	/com/las
2D0000 - 2F7FFF	B0000	-- temporary file --
BC0000 - BCFFFF	0	/help_area/worksite
BD0000 - BDFFFF	0	/jttj

1408 KB mapped.

3.

```
$ las -f 188000 -t 200000
```

VA Range	Obj Start	Pathname
188000 - 19FFFF	0	/lib/tfp
1A0000 - 1BFFFF	0	/lib/dialoglib
1C0000 - 1C7FFF	0	/sys/node_data/ipc_data
1D0000 - 1D7FFF	30000	/sys/node_data/global_data

288 KB mapped.

NAME

lb_admin – Location Broker Administrative Tool

SYNOPSIS

`/etc/ncs/lb_admin`

DESCRIPTION

The **lb_admin** tool monitors and administers Location Broker registrations. It presents both a Domain/Dialogue (tm) based user interface and a terminal-oriented interface. For information about the Domain/Dialogue interface, use the **HELP** key while running the tool. Information about individual commands for the terminal-oriented interface is available through the **help** command.

COMMANDS

help	List available commands or get information about a specific command.
quit	Exit the lb_admin session.
lookup	List matching Location Broker entries.
register	Add an entry to the Location Broker database.
set_broker	Select the specific Location Broker to use.
use_broker	Direct operations to a Local Location Broker or to a Global Location Broker.
unregister	Delete an entry from the Location Broker database.

NAME

lbr – combine object modules into a library

SYNOPSIS

lbr {**-c** | **-upd**} *library_pathname* [*module_pathname*] [*options*] [-]

DESCRIPTION

The **lbr** command manages libraries of object modules. It adds, removes, or replaces modules in the library. As input, you must provide the pathname of a library you want to create or update, followed by an optional list of object module pathnames and processing options. As output, **lbr** produces a new or updated library file.

You can use **lbr** in two ways: by entering complete **lbr** command strings, or by using the "-" (hyphen) option to ask **lbr** to prompt you for multiple strings of *module_pathname* arguments and options. By using prompting you can perform several operations on object modules in the same library file, without entering **lbr** each time.

For a complete description of **lbr**, see the *Domain/OS Programming Environment Reference*.

Prompting

The optional hyphen at the end of the command line requests **lbr** to begin prompting. The hyphen is valid only on the line containing the **lbr** command, and must be the last item on the line. Note that prompting is also requested if the command line contains only the **lbr** command.

If you request prompting, **lbr** processes the arguments and options on the current command line, then displays an asterisk (*) on standard output. In response to the asterisk, you can enter additional *module_pathname* arguments and options. For example,

```
$lbr -c mylib.lib
*file1.bin -del my_module
*file2.bin -l -repl file3.bin
*
```

Prompting ends when you specify the **-end** option or press RETURN in response to the asterisk. After prompting ends, **lbr** finishes creating or updating the library file.

Comment Statements

You can include comments to an **lbr** command during a prompting session or in a shell script. Comments must be delimited by braces, as in this example:

```
$lbr -upd plot.lib
*plot_line.bin { Add plot_line procedure to library }
*{ Generate library directory }
*_1
*-end
```

lbr ignores any comments when it processes the command line.

Librarian Errors

If a problem occurs during **lbr** execution, **lbr** displays a message on error output. The message indicates the nature and severity of the problem. Error-level messages are issued for fatal conditions, which prevent **lbr** from creating or updating a library file. Warning-level messages are issued for conditions that do not prevent **lbr** from producing a library file, but the file's contents may not be what you expect.

ARGUMENTS

-c[reate] | -upd[ate] library_pathname (required)

The pathname of the library output file must be specified on the command line before you can specify any option that performs an operation on a library (such as adding to, extracting from, or reporting about a library). The **-c** (create) or **-upd** (update) option must be specified with the library pathname argument to indicate whether you want to create or update a library. Remember that only one library output file can be specified per execution of **lbr**.

module_pathname (optional)

Specify an object module to be added to the library. Multiple pathnames and wildcarding are permitted. If omitted, no new object modules are added to the library.

OPTIONS

The following options instruct the librarian to perform various tasks. Note that some options apply directly to a library, while others act on modules within the library.

-del module Remove an object module from the library. If a module of the given name cannot be found in the library, a warning is issued. Note that the librarian is case-sensitive to the name *module*.

-ex module [-o pathname] Extract the named module from the library. If the pathname modifier is specified with **-o**, the module is copied to that pathname. Otherwise, the module is copied to a file having the same

- name as the module. Note that the librarian is case-sensitive to the name *module*.
- l List a directory of the library contents to standard output.
 - msgs (default) Cause **lbr** to issue purely informational messages such as a summary of the number of errors and warnings that occurred.
 - nmsgs Cause **lbr** to suppress issuing purely informational messages.
 - repl *pathname* Replace, in the library, any modules found in the file specified by *pathname*. This option has an effect equivalent to first deleting all the modules found in *pathname* from the library, and then adding all the modules in *pathname* back into the library. The advantage gained by using **-repl** is that you do not need to know the names of the modules in *pathname*. Also, if you attempt to add a module to a library without using the **-repl** option, and a module of that name already exists, an error message is issued. If a module found in *pathname* does not already exist in the library, a warning message is issued.
 - (hyphen alone) Request librarian prompting for further arguments.

NAME

lbr2ar – convert lbr libraries to SR10 archive libraries

SYNOPSIS

lbr2ar [*-y dirname*] *lbrfile arfile*

DESCRIPTION

The **lbr2ar** command converts pre-SR10 lbr library files containing object modules in OBJ format to SR10 ar library archive files containing object modules in COFF format. The **lbr2ar** command extracts each object module from the *lbrfile*, executes the **obj2coff** converter to convert them to COFF, and creates a library archive (*arfile*) containing the converted object modules. Note that both the library format and the format of the individual object modules are changed.

OPTIONS

-y dirname This option allows you to specify a new pathname, *dirname*, for the location of **obj2coff**. The new pathname for **obj2coff** is *dirname/obj2coff*. The default pathname for **obj2coff** is */usr/apollo/bin*.

FILES

<i>/usr/apollo/bin/obj2coff</i>	obj2coff converter
<i>/tmp/obj/*</i>	Temporary files
<i>/tmp/coff/*</i>	Temporary files

SEE ALSO

More information is available. Type

help obj2coff	For information on converting OBJ format modules to COFF format modules
----------------------	---

NAME

lcm – load a color map

SYNOPSIS

lcm [**-p** *pathname*]

DESCRIPTION

lcm loads a color map from a file that specifies a set of color map entries. Each entry establishes an association between an index and a color value. When the DM is initially loaded, it sets the node's color map from the file in `/sys/dm/color_map`.

If no *pathname* is given, **lcm** loads the color map from `/sys/dm/color_map`. In this case, all 16 colors (that is, color entries for color slots 0-15) are reloaded. If you specify a *pathname*, **lcm** reads the given file and tries to load the colors associated with the indexes.

NOTE

If there are direct mode graphics programs running that have changed the color values for color slots 0-15, the execution changes the colors in these windows as well as resetting the DM's colors.

OPTIONS

-p *pathname* Specify the file that contains the color values for red, green, and blue. The format of this file should be identical to the DM's color map file, `/sys/dm/color_map`. For more information about the format of this file, please refer to the manual *Programming with Domain Graphics Primitives*.

EXAMPLES

Load the DM's color map found in the file `/sys/dm/color_map`.

```
§ lcm
```

Load the color map specified in the file `my_colormap`.

```
§ lcm -p my_colormap
```


NAME

lcnet – display internet routing information

SYNOPSIS

/etc/lcnet [options]

DESCRIPTION

lcnet displays the list of known networks, their distance from the current node, the router used as the first hop from that network, and other information.

The distance (hops) from remote networks is measured in intervening routers. The distances are all for one-way traffic; if a network is three hops away from yours, your requests pass through three routers to get to that network. The responses also pass through three routers on the way back.

The **-conn** option shows you the full internet topology; that is, the list of networks and how the routers connect them together.

OPTIONS

-local (default)

Display the "First Hop" and "Hops" information for each network in the internet. The first hop is the node ID of a router on your network. It is the first router used in sending packets from your network to the target network. Other routers are also used if the target network is more than one hop away from your own.

-full

Display information showing how up to date the routing table is (the "Age" and "Expiration" columns) in addition to the "First hop" and "Hops" information shown by the **-local** option. **-full** also lists inaccessible networks.

-conn

Show which routers are connected to each network, and which other networks those routers touch. This option displays the "Touching" information.

-hw

Display the type of hardware used for each of the networks (ring or IIC).

The **-conn** and **-hw** options may take several seconds to execute if you have a large internet.

-n node-spec

Print another node's view of the internet. The outputs produced by **-local** and **-full** vary from node to node; **-n** affects these outputs. The **-n** option does not affect the output produced by the **-conn** or **-hw** options, since the hardware and connectivity do not depend on a node's position in the internet.

- c** The **-c** option suppresses the title over each output column. It also fills every line of the "Network" column produced by the **-conn** option, and every line of the "Hardware" column produced by the **-hw** option. These format changes make it easier to use **lcnet**'s output as another program's input.

EXAMPLES

In this example, the node is directly connected to network **COFFEE**. Networks **5A1AD** and **ED1F1CE** were connected in the past, but are not now accessible (perhaps because the routers are down).

The expiration date and time for the "local" network are meaningless.

```
$ /etc/lcnet -full
```

Network	First Hop	Hops	Age	Expiration date/time
=====	=====	=====	===	=====
B020	4B6F	1	NEW	1987/06/16 14:33:21
B00B00	4B6F	2	NEW	1987/06/16 14:33:21
5A1AD	4B6F	gone	NEW	1987/06/16 14:33:21
COFFEE	0	local	NEW	1987/06/09 10:27:46
ED1F1CE	4B6F	gone	NEW	1987/06/16 14:33:21
D0D0	BAD1	1	NEW	1987/06/16 14:33:39

The "Touching" information describes your internet completely. This example includes several kinds of information:

- Network **DEFACED** has one router, node **2A3B**.
That router connects **DEFACED** to **EFFACED**.
- Network **FACE0FF** contains two routers, **31DC** and **1371**.
Those routers connect **FACE0FF** to **C0C0A** and **C0FFEE**, respectively.

\$ /etc/lcnet -conn

Network	Touching Router	Touching Network
=====	=====	=====
F00D	5C0B	DECAF
	36CF	COFFEE
5A1AD	459B	COFFEE
	45BE	ED1F1CE
B002E	3F0A	COFFEE
C0C0A	BAD1	B00B1E
	56B0	EFFACED
	31DC	FACE0FF
DECAF	5C0B	F00D
B00B1E	BAD1	C0C0A
COFFEE	36CF	F00D
	459B	5A1AD
	3F0A	B002E
	1371	FACE0FF
DEFACED	2A3B	EFFACED
ED1F1CE	45BE	5A1AD
EFFACED	56B0	C0C0A
	2A3B	DEFACED
FACE0FF	31DC	C0C0A
	1371	COFFEE

\$ /etc/lcnet -conn -c

F00D	5C0B	DECAF
F00D	36CF	COFFEE
5A1AD	459B	COFFEE
5A1AD	45BE	ED1F1CE
B002E	3F0A	COFFEE
C0C0A	BAD1	B00B1E
C0C0A	56B0	EFFACED
C0C0A	31DC	FACE0FF
DECAF	5C0B	F00D
B00B1E	BAD1	C0C0A
COFFEE	36CF	F00D
COFFEE	459B	5A1AD
COFFEE	3F0A	B002E
COFFEE	1371	FACE0FF
DEFACED	2A3B	EFFACED
ED1F1CE	45BE	5A1AD
EFFACED	56B0	C0C0A

LCNET

Aegis

LCNET

EFFACED	2A3B	DEFACED
FACEOFF	31DC	C0C0A
FACEOFF	1371	C0FFEE

SEE ALSO

More information is available. Type

help lcnode

For information on listing connected nodes

NAME

lcnode – list nodes connected to the network

SYNOPSIS

/etc/lcnode [*options*]

DESCRIPTION

lcnode lists the nodes currently connected to the network. The list contains the ID of every node connected, the time at which the node was started, the current time, and the name of each node's entry directory.

This command reports only the nodes that respond within a preset time limit. If a node is connected, but temporarily unable to respond within the specified time, it does not appear in the produced list.

OPTIONS

- m[e]** Request information about your node only. This option displays the node ID.
- b[rief]** Request brief output. **lcnode** lists only the entry directory name for each connected node. Note that the entry directory of a diskless node is the entry directory of its paging partner.
- id** When used with **-brief**, display the node ID in addition to the entry directory.
- c[ount]** Request node count only. **lcnode** lists only the number of nodes responding to its query.
- max[nodes] n** Set a limit on the number of nodes you want to see, even if more could respond.
- from node_spec** Starts the node list at some node other than your own. This is especially useful in an internet environment, for looking at networks other than your own. See **help node_spec** for details about node specification syntax.
- name** When you specify the **-brief** option, **lcnode** normally prints the entry directory for each node. If you specify **-name** with **-brief**, **lcnode** prints the node name cataloged with the naming server. Only diskless nodes are printed differently. A diskless node's entry directory is its partner's node name; a diskless node's node name is uniquely its own.

Unless the **-from** option specifies your own node, the list includes only an unbroken sequence of nodes running Aegis SR9.0 or later. The rest of the node list is lost, starting with the first node running a pre-SR9.0 Aegis.

EXAMPLES**1. \$ /etc/lcnode**

The node ID of this node is 21. 3 other nodes responded.

id	Boot time	Current time	Entry Directory
21	1987/06/09 9:21:44	1987/06/09 16:06:22	//dollar
17	1987/06/09 13:52:02	1987/06/09 16:06:13	//quarter
27	1987/06/09 12:53:28	1987/06/09 16:06:07	//nickel
11	1987/06/09 12:03:39	1987/06/09 16:06:15	** DISKLESS ** //diskless_\$11 partner node: 17

2. \$ /etc/lcnode -me

The node id of this node is 21.

3. \$ /etc/lcnode -b

```
//dollar
//quarter
//nickel
//quarter
```

(//quarter appears once as the host for a diskless node and once for the node with the disk.)

4. \$ /etc/lcnode -b -name

```
//dollar
//quarter
//nickel
//diskless_$000011
```

(-name shows you the name under which diskless node 11 is cataloged)

5. \$ /etc/lcnode -c

466 other nodes responded.

6. \$ /etc/lcnode -c -b

466

7. \$ /etc/lcnode -c -m

The node id of this node is 116A.
466 other nodes responded.

8. \$ /etc/lcnode -b -id

21 //dollar
17 //quarter
27 //nickel
11 //quarter

9. \$ /etc/lcnode -from 0FAD.3924 -max 2

Starting from node 3924.

1 other node responded,

but more might have responded with a high -max value.

Node id	Boot time	Current time	Entry Directory
3924	1985/02/14 17:20:45	1985/02/14 19:07:04	//laurel
34Bf	1985/02/14 18:46:52	1985/02/14 19:08:09	//hardy

SEE ALSO

More information is available. Type

help lcnet

For information on listing connected networks in an internet environment

NAME

ld – list contents of a directory

SYNOPSIS

ld [*pathname...*] [*options*]

DESCRIPTION

ld lists the objects in a directory on standard output. It provides a wide variety of information on the contents of the various objects, depending on the command options you select.

ARGUMENTS

pathname (optional)

Specify *pathname* of the object to be described. The object may be a directory, a file, or a link. If you specify a directory, **ld** describes the files in that directory. If you specify a file, **ld** describes attributes of that file. Multiple *pathnames* and wildcarding are permitted. (If they are used, each name is assumed to be a filename.)

Default if omitted: list contents of working directory

OPTIONS**Attributes**

- a** Display all attributes.
- attr** Display permanent/immutable/trouble flags.
- bl** Display disk blocks used.
- len** Display current length in bytes.
- r** Display your access rights to entries.
- root** Display the contents of the replicated root directory managed by the naming server helper.
- st** Display system object type.
- tu** Display type UIDs.

Date and Time

- d** Display creation, modified, and last-used dates.
- dtc** Display date/time created.
- dtm** Display date/time last modified.
- dtu** Display date/time last used.

Streams

- si** Display all stream header information.
- ab** Display streams ASCII/binary flag.
- conc** Display streams object concurrency.
- rt** Display streams record type.

Entry Selection

- crb *d*** Display entries created before date and time *d*.
- cra *d*** Display entries created after date and time *d*.
- usb *d*** Display entries used before date and time *d*.
- usa *d*** Display entries used after date and time *d*.
- mob *d*** Display entries modified before date and time *d*. Same as old **-be** option.
- moa *d*** Display entries modified after date and time *d*. Same as old **-af** option.
- be *d*** Display entries modified before date and time *d*. Obsolete option: use **-mob**.
- af *d*** Display entries modified after date and time *d*. Obsolete option: use **-moa**.
- di** Treat all names as directory names and list the contents of those directories.
- ent** List attributes of the target object itself. This option has no effect if the pathname refers to a file. If the target object is a directory, **-ent** causes **ld** to display attributes of the directory itself rather than its contents. If the target object is a link, **-ent** causes **ld** to display attributes of the link itself rather than trying to resolve the link and display attributes of the resolution object. See Example 5 below.
- ld (default)** List directory names. If this option is specified, then **-lf**, **-ll**, and **-ln** lose their default status, and must be specified explicitly, if desired.
- lf (default)** List filenames. If this option is specified, then **-ld**, **-ll**, and **-ln** lose their default status, and must be specified explicitly, if desired.
- ll (default)** List link names. If this option is specified, then **-ld**, **-lf**, and **-ln** lose their default status, and must be specified explicitly, if desired.
- ln (default)** List diskless node names. If this option is specified, then **-ld**, **-lf**, and **-ll** lose their default status, and must be specified explicitly, if desired. Diskless node names normally appear only when you specify **-root**, or when you list the **//** directory.
- lt** Display link resolution names.

Output Control

- sc** Sort the output vertically in columns.
- sr (default)** Sort the output horizontally in rows.
- w *n*** Adjust the output to be *n* characters wide. If this option is omitted, **ld** automatically adjusts the width of the output to the size of the transcript pad's window, unless the command is issued from a dumb terminal or some other windowless device. In that case, the output defaults to 80 characters wide if **-w** is omitted.
- c** List entries in a single column, suppress header.
- hd (default)** Display header and totals.
- nhd** Suppress header and totals.
- sn (default)** Sort entries by name.
- nsn** Suppress entry sorting.
- warn (default)** Produce a warning if no wildcard matches are found.
- nwarn** Suppress warning if no wildcard matches are found.
- h[idden]** Names the directories "." (the current working directory) and ".." (the parent directory); these always appear first, even when a sort flag is on.

ld uses the command-line parser, and so also accepts the standard command options with the exception of the query options (**-qa**, **-nq**, **-qw**). Type **help cl** for more information.

Time

The time at which a file is created, modified, or used is accurate within a certain tolerance. The reported time of creation or modification is correct within one minute of the actual creation or modification time. The time of last use is updated only if more than an hour has elapsed since the recorded time of last use. Hence, the time of last use reported by the **ld** command may vary by as much as an hour from the actual time of last use.

EXAMPLES

1.

\$ ld -a

Directory "/col/users/final1":

sys	type	blocks	current				
type	uid	used	length	attr	rights	name	
file	rec	18	17640	P	pndwrx	ch1	
file	rec	18	18428	P	pndwrx	ch2	
file	rec	67	67210	P	pndwrx	ch3	
file	rec	12	11554	P	pndwrx	ch4	

4 entries, 115 blocks used.

2.

\$ ld -dtm

Directory "/col/users/final1":

date/time	modified	name
88/09/28 17:18		ch1
88/09/28 17:18		ch2
88/09/28 17:19		ch3
88/09/28 17:20		ch4

4 entries, 115 blocks used.

3.

\$ ld /sys/ins/[a-e]?*.ins.ftn -a

sys	type	blocks	current				
type	uid	used	length	attr	rights	name	
file	rec	1	872	P	pndwrx	/sys/ins/base.ins.ftn	
file	rec	2	1274	P	pndwrx	/sys/ins/cal.ins.ftn	

```
file unstruct      20   19966 P   pndwrx  /sys/ins/core.ins.ftn
file rec           1     738 P   pndwrx  /sys/ins/ec2.ins.ftn
```

4 entries listed, 24 blocks used.

4. In this example, `//victor` is the name of a diskless node.

```
$ ld //v?* -a
```

sys	type	blocks	current				
type	uid	used	length	attr	rights		name
node							//victor
sdir	nil	5	5120	P	-----rse		//visitor
	(attributes unavailable)						//void
sdir	nil	3	3072	P	pgn-calrse		//vulture

4 entries listed, 8 blocks used.

5.

This example produces an error because the resolution object `//behemoth/rkd/foo.dat` does not exist. Use the `-ent` to show attributes of the link itself without trying to resolve it.

```
$ curl foo //behemoth/rkd/foo.dat
```

```
$ ld foo -ll -lt
```

```
?(ld) "foo" - name not found (os/naming server)
```

```
$ ld foo -ll -lt -ent
```

```
foo "//behemoth/rkd/foo.dat"
```

1 entry listed.

6.

The following command displays the contents of the working directory and displays attributes of the working directory itself.

```
$ ld.-a
```

```
Directory "//otis/tstlib/trash":
```

sys	type	blocks	current				
type	uid	used	length	attr	rights		name
file	unstruct	1	32	P	pgndwrx		abc
link							foo

```
2 entries, 1 block used.
```

```
$ ld.-ent-a
```

sys	type	blocks	current				
type	uid	used	length	attr	rights		name
dir	nil	2	2048	P	pgndcalrse		

```
1 entry listed, 2 blocks used.
```

SEE ALSO

More information is available. Type

help datetime For information on date-time syntax

NAME

lkob – lock an object

SYNOPSIS

lkob *pathname* [-r|-w|-i|-r2w|-r2riw|-w2r|-w2riw]

DESCRIPTION

lkob locks the specified object. The locking constraint is "n readers xor 1 writer".

lkob is primarily used for system-level debugging.

Use **llkob** (`list_locked_objects`) to list locked objects. Use **ulkob** (`unlock_object`) to unlock an object.

ARGUMENTS

pathname (required)

Specify object to be locked. Multiple pathnames and wildcarding are permitted.

OPTIONS

- r** (default) Lock the object for reading.
- w** Lock the object for writing.
- i** Lock the object for reading, with intent to write.
- r2w** Change the lock mode of the object from "read" or "read-intend-write" to "write".
- r2riw** Change the lock mode of the object from "read" to "read-intend-write".
- w2r** Change the lock mode of the object from "write" to "read".
- w2riw** Change the lock mode of the object from "write" to "read-intend-write".

This command uses the command-line parser, and so also accepts the standard command options listed in `help cl`.

EXAMPLES

```
§ lkob susan -w
```

Lock file `susan` for writing.

SEE ALSO

More information is available. Type:

- help llkob** For details about listing locked objects
- help ulkob** For details about unlocking locked objects

NAME

llbd – Local Location Broker Daemon

SYNOPSIS

`/etc/ncs/llbd`

DESCRIPTION

The Local Location Broker Daemon (**llbd**) is part of the Network Computing System (NCS). It manages the Local Location Broker (LLB) database, which stores information about NCS-based server programs running on the local node.

A host must run **llbd** if it is to support the Location Broker forwarding function or to allow remote access (e.g., by the `lb_admin` tool) to the LLB database. In general, any node that runs an NCS-based server program should run an **llbd**. Additionally, any network supporting NCS activity should have at least one node running the Global Location Broker Daemon (**glbd**). Typically, both daemons are started at boot time from the `/etc/rc` file.

To start **llbd** on a node that is already running, type the following at a shell prompt:

```
§ /etc/server/etc/ncs/llbd &
```

To have **llbd** start every time a node boots, use `touch` or `crf` to create the file `/etc/daemons/llbd`.

If **llbd** is to support remote access from hosts in the IP address family, a TCP daemon (`tcpd`) must be running on the local node; `tcpd` should be started before **llbd**.

NAME

llib – list installed libraries

SYNOPSIS

llib [**-a**]

DESCRIPTION

The **llib** command lists those libraries which have been installed in the current process via the build-in **inlib** shell. These libraries are used to resolve unknown references when loading a program. To find out if a symbol is known and will be used in resolving an unknown reference, use **esa**.

OPTIONS

-a Also list those libraries which are known globally to every process. These libraries are installed at boot time using the configuration information in **/etc/sys.conf**.

NAME

llkob – list locked objects

SYNOPSIS

llkob [*options*]

DESCRIPTION

llkob lists the locked objects resident on volumes mounted on this node, and objects resident in other nodes that are locked by processes running locally.

The listing for each object includes the locking constraints imposed on the object (for example, n-readers XOR 1-writer), the specific lock mode being used (for example, read, write, read-intending-write), the network node ID of the node at which the object is located, the node ID of the node in which the locking process is active, and the name (if it is available) of the object itself.

OPTIONS

- r[emote]** Specify list of only those objects that either reside on this node and are locked by another node, or reside on another node and are locked by this node (that is, those objects whose locks are in some way remote).
- c[ount]** List only a one-line summary of the number of objects locked.

EXAMPLES

\$ llkob

USE	CONSTRAINT	HOME NODE	LOCKING NODE	FILE
W	nR_xor_1W	21	21	/sys/dm/pdb
R	nR_xor_1W	21	21	/sys/dm/fonts/std
W	nR_xor_1W	21	21	--Temporary File--
R	nR_xor_1W	21	21	--Uncataloged Permanent File--
W	nR_xor_1W	21	21	--Display Manager Pad--

\$ llkob -c

locked: 102 -- 100 local, 2 remote; 100 locally locked, 2 remotely

SEE ALSO

More information is available. Type

- help lkob** For details about locking objects
- help ulkob** For details about unlocking locked objects

NAME

lopstr – list open streams

SYNOPSIS

lopstr

DESCRIPTION

lopstr lists the streams that are open for the current process. The list contains the stream ID and access mode (read, write, append, and so forth) for each stream. The pathname (if one exists) associated with each stream is also displayed.

lopstr requires no arguments or options.

EXAMPLES

\$ lopstr

st#	open	name
0	read	(standard input)
1	append	(standard output)
2	read	(error input)
3	append	(error output)

4 streams open.

NAME

lprotect – control local protection

SYNOPSIS

/etc/lprotect [**-e rootlocal**] [**-d rootlocal**]

DESCRIPTION

The **lprotect** command controls local protection attributes on a node. Currently, this command enables requests by root (locksmith) to be honored only if they originate locally (**rootlocal**), i.e. from a local process. If no options are specified, the current state of **rootlocal** is returned. To change the state of the **rootlocal** attribute, you must be running as root (locksmith).

OPTIONS

-e rootlocal Enables local-only root requests.
-d rootlocal Disables local-only root requests.

EXAMPLE

1. Show current status.

```
$ /etc/lprotect  
"local-only root requests" is disabled. (-d rootlocal)
```

2. Enable local root requests

```
$ /etc/lprotect -e rootlocal  
$ /etc/lprotect  
"local-only root requests" is enabled.
```

3. Disable local root requests

```
$ /etc/lprotect -d rootlocal
```

NAME

lst – list contents of a storage tree

SYNOPSIS

lst *source* [*options*]

DESCRIPTION

lst prints the number of kilobytes contained in all files and directories specified by names. Links within a tree are not followed. If you do not specify a names argument, **lst** prints information about the current directory by default. Temporary file space is not included in this count.

OPTIONS

- ae** Abort on an error. The default is to continue the list.
- af *date*** List files with a date/time modified greater than specified date/time.
- be *date*** List files with a date/time modified less than specified date/time.
- l** List all directories, files, and links.
- ld (default)** List directories.
- lf** List files.
- ll** List links.
- dtm** List the date/time modified of objects selected.
- lev *n*** List only objects *n* levels or fewer below source directory.
- st** List statistics.
- nsd** Ignore system directories (that is, mounted file systems).

lst traverses the specified tree and counts storage occupied by files satisfying the optional date criteria, accumulating totals for every directory. Links within the tree are not followed.

-ld, **-lf**, **-ll** may be negated as **-nld**, **-nlf**, **-nll**.

Wildcards or link may be specified for the source pathname. This command uses the common command-line handler; type **help cl** for more information.

Statistics reports the following information per nest level for each tree:

- no. of files, storage utilization for the files
(in kilobytes),
storage/file
- no. of trees, storage utilization for the trees

(in kilobytes),
 storage/tree
 - no. of links

If more than one tree is specified with the `-st` option the following is generated:

- individual statistics report for each tree
- average statistics report over all specified trees

Note that `trees = 0` with some value for storage reflects the overhead for that directory's files.

EXAMPLES

`$ lst backup/performance backup/backup.info -lev 1 -st`

(dir) 1002 backup/performance

Tree Statistics for backup/performance

nest	files	storage		trees	storage		links
		storage	/ file		storage	/ tree	
0	34	998	29.4	0	4	0.0	2
	-----	-----		-----	-----		-----
	34	998	29.4	0	4	0.0	2

(dir) 13 backup/backup.info/debug_info

(dir) 45 backup/backup.info/bug_info

(dir) 19 backup/backup.info/info

(dir) 279 backup/backup.info

Tree Statistics for backup/backup.info

nest	files	storage		trees	storage		links
		storage	/ file		storage	/ tree	
0	28	199	7.1	3	80	26.7	0
1	19	73	3.8	0	4	0.0	3
	-----	-----		-----	-----		-----
	47	272	5.8	3	84	28.0	3

Averages for 2 trees

nest	files		storage		storage / file
	ave	stdev	ave	stdev	
0	31.0	3.0	598.5	399.5	19.3
1	9.5	9.5	36.5	36.5	3.8
	-----	-----	-----	-----	
	20.3	6.3	317.5	218.0	15.7

LST

Aegis

LST

trees	trees	storage	storage	storage	links	links
ave	stdev	ave	stdev	/ tree	ave	stdev
1.5	1.5	42.0	38.0	28.0	1.0	1.0
0.0	0.0	2.0	2.0	0.0	1.5	1.5

0.8	0.8	22.0	20.0	29.3	1.3	1.3

SEE ALSO

More information is available. Type

help datetime

For information on date-time syntax

NAME

lty – list installed types

SYNOPSIS

lty [*options*]

DESCRIPTION

lty lists the types currently installed on a volume. It can also be used to list the contents of internal caches for debugging purposes.

OPTIONS

If no options are specified, **lty** lists types installed on the boot volume.

- n *node_spec*** Specify the node whose type names are to be listed. Type **help node_spec** for details about node specification syntax. You may also specify the entry directory of a volume mounted for software installation, as shown in the example below.
- u** Display type UIDs as well as type names.
- glob** Display contents of global type name cache instead of the type file (for debugging only).
- priv** Display the contents of the private (per-user) type name cache instead of the type file (for debugging only).

EXAMPLES

\$ lty

Local type file

```
area  bitmap  boot  casehm  ddf  evetype  hdru  ipad
lheap mbx     mt    nil     null  obj     objlib pad
pipe  rec     sch   sio     uasc  und
```

In the following example, the disk has been mounted for software installation. The disk's top level directory (cataloged as **/mounted_disk** by the **mtvol** command) must contain a **sys** directory. If it does not, you get a "types file not found" error.

\$ mtvol w /mounted_disk

\$ lty -n /mounted_disk

Type file for "//my_node/mounted_disk"

```
area  bitmap  boot  casehm  ddf  evetype  hdru  ipad
lheap mbx     mt    nil     null  obj     objlib pad
pipe  rec     sch   sio     uasc  und
```

LTY

Aegis

LTY

SEE ALSO

More information is available. Type

help crty For information on creating types

help dlty For information on deleting types

NAME

lusr – list logged on users

SYNOPSIS

lusr [*options*]

DESCRIPTION

lusr lists the identities of active users on the network.

OPTIONS

If no options are specified, the person name and node entry directory of the user logged into the DM is listed.

- me** List the user logged on to this node by person, group, organization name, and node ID.
- all[nodes]** List all nodes the user is logged on to by person, group, organization name, and node ID.
- n *node_spec* ...** Lists user(s) logged on to the node specified. See **help node_spec** for details about node specification syntax. Multiple pathnames or node IDs are permitted; separate them with blanks.
- br** Suppress listing of home directory names. Home directory names are listed if this option is not specified.
- full** List complete PGON (person, group, organization name, and node ID) for each user listed.
- nofull** List only the person name of each user listed.
- allp[rocs]** List identities for all user processes, not just the DM, by node (if **-n** is also specified), by name (**-pgo**), for the current node only (**-me**), or everywhere in the network.
- pgo *pgo*** List user(s) named, at all nodes from which they have logged in to the DM. *pgo* is a string of the form *pers.group.org*, where '%' may be used as a wildcard specifier and trailing %'s may be omitted (for example, *%.os_dev* or *joe.%r_d*).
- idle** Include idle nodes in report. If you omit this option, **lusr** suppresses the names of nodes at which no one is logged in.

EXAMPLES

```
$ lusr -me
  loc.none.mfg.1D5      //et
$

$ lusr -me -nofull -br
  loc
$
```

```
$ lusr -n //magic //mountain //park
```

```
  joe           //magic
  brian         //mountain
  gordon        //park
```

```
$
```

```
$ lusr -full
```

```
jack.none.none.532      //zoo
andy.none.now.12B      //me
carol.none.mtg.334     //vip
nelson.none.pres.838   *** diskless 383 ***
                       //halfwit partner node: //plan
annie.none.r_d.6CA     //lunar
now.system.advent.368  *** diskless 368 ***
                       //diskless_$000368 partner node: //zoid
beth.none.mfg.2F7     //mack
```

```
$
```

```
$ lusr -idle
```

```
joe           //magic
*No one logged in* //stride
*No one logged in* //panacea
janet        *** diskless //lala ***  partner node: //nirvana
john         //duck
eric         //lion
*No one logged in* //fourbits
harp         //polo
```

```
$
```

NAME

lvar – list information about set variables

SYNOPSIS

lvar [*var_name* ...]

DESCRIPTION

The **lvar** command lists the type, name, and value of currently set variables. Optionally, you can specify individual variable names.

ARGUMENTS

var_name ... (optional) List type, name, and value of the specified variable(s).

Default if omitted: list information for all variables currently set

NAME

lvofls – list free space on logical volumes

SYNOPSIS

lvofls [*pathname*] [*options*]

DESCRIPTION

lvofls prints information about the amount of available storage on mounted volumes. This information includes the total amount of storage in disk blocks, the amount of free storage, the percentage of the total storage that is free, and the entry directory name for the volume.

ARGUMENTS

pathname (optional) Report on the volumes mounted on the home node of the specified file.

Default if omitted: list free space on current node

OPTIONS

If no options are specified, **lvofls** reports the storage available on the volumes mounted on the current node.

- a** Report on all volumes mounted in the network.
- n *node_spec* ...** Report on the volumes mounted on the specified node[s]. Multiple *node_spec* strings are permitted; separate them with blanks.

EXAMPLES

\$ lvofls -a

# free	# total	% free	node id	entry directory
24217	30012	81	1A	/
16589	30012	55	2B	//dev
7927	30012	26	3C	//lang
14497	30012	48	4D	//mkt

SEE ALSO

More information is available. Type

help node_spec For details about node specification syntax

NAME

macro – expand macro definitions

SYNOPSIS

macro [-0] [*pathname* ...]

DESCRIPTION

macro is a general purpose macro processor. **macro** reads the files and writes to standard output a new file with the macro definitions deleted and the macro references expanded.

Macros permit the definition of symbolic constants so that subsequent occurrences of the constant are replaced by the defining string of characters. The general form of a macro definition is

define(*name*,*replacement text*)

All subsequent occurrences of *name* in the file will be replaced by *replacement text*. The string *name* can consist of letters (a-z and A-Z), digits (0-9), underscores (`_`), and dollar signs (`$`). The placement of blanks in definitions is significant; they should only appear in the replacement text where desired. Uppercase and lowercase letters are also significant. The replacement text may be more than one line long. However, when an entire macro definition is followed immediately by a newline, the newline is discarded. This prevents extraneous blank lines from appearing in the output.

Macros with arguments may also be specified. Any occurrence in the replacement text of *\$n*, where *n* is between 1 and 9, will be replaced by the *n*th argument when the macro is actually called. No space is allowed between the command (in this case, **define**), and the left parenthesis.

ARGUMENTS

pathname (optional)

Specify file containing macro definitions to be processed. Multiple pathnames are permitted.

Default if omitted: read standard input

OPTIONS

-0 (Zero, not letter O)

Remove one level of brackets in macro calls prior to processing. Normally, brackets appearing outside any macro calls (level zero brackets) are not removed.

Built In Macros

The following built-in macros are provided:

define(*a*,*b*) Defines *a* to be *b* and returns the null string.

ifelse(*a*,*b*,*c*,*d*) Returns *c* if *a* is identical to *b*. Otherwise, it returns *d*.

incr(*a*) Interprets *a* as an integer and returns *a+1*.

substr(*a,m,n*) Returns a substring of the string *a* starting at character number *m* and extending for *n* characters.

len(*a*) Returns the length of *a*.

includ(*a*) Returns the contents of file *a*.

expr(*a*) Returns the result of evaluating infix expression *a*. Operators in increasing order of precedence are as follows. Parentheses may be used as usual.

&	Logical OR and AND
!	Unary logical NOT
== ^= <= < > >=	Arithmetic comparison
+ -	Addition and subtraction
* / %	Multiplication, division, modulus (remainder)
**	Exponentiation
+ -	Unary plus and negation
Logical operators return 0 (false) or 1 (true)	

EXAMPLES

A simple example of a macro is

```
define (EOF, -1)
```

Thereafter, all occurrences of EOF in the file are replaced by '-1'.

You may specify arguments in macro definitions with the characters *\$n*, where *n* is a number between 0 and 9. The arguments to be inserted when the macro is encountered are given inside parentheses following the macro name. \$0 refers to the name of the macro itself. For example,

```
define (copen, $3 = open ($1, $2) )
```

defines a macro that, when called by

```
copen (name, read, fd)
```

expands into

```
fd = open (name, read)
```

If a macro definition refers to an argument that was not supplied, the *\$n* is ignored. The \$ is taken literally if a character other than a digit follows it.

Macros can be nested, and can be called recursively. Any macros encountered during argument collection are expanded immediately, unless they are surrounded by square brackets ([]). That is, input surrounded by brackets is left alone, except that one level of [and] is stripped off. Thus it is possible to write the macro *d* as

```
define (d, [define ($1, $2) ])
```

The replacement text for *d*, protected by the brackets, is literally 'define(\$1,\$2)' so you could use:

```
d(a, bc)
```

to define *a* as *bc*. Brackets must also be used to redefine a macro. For example

```
define (x, y)
.
.
.
define (x, z)
```

defines *y* in the second line, instead of redefining *x*. To define *x* the second time, the operation must be expressed as

```
define (x, y)
.
.
.
define ([x], z)
```

Normally, brackets appearing outside any macro calls (level zero brackets) are not removed. When the -0 (zero, not letter O) option is specified, one level of brackets is removed both inside and outside the macros. One level of brackets is also removed when the macro reference is expanded. Thus, to rewrite the *d* macro above so that it is evaluated to the literal string 'define(\$1,\$2)', the definition is

```
define (d, [[define ($1, $2) ]])
```

In order to redefine the macro `define` (for example, so that the Pascal keyword 'define' can be used) the following definition can be used:

```
define ([define], [[define]])
```

One level of brackets is stripped from both arguments when the definition is processed.

The second argument is stripped when the macro is invoked.

DIAGNOSTICS

arith evaluation stack overflow

Arithmetic expressions can be nested only to 30 deep.

arg stack overflow

The total number of arguments exceeds the limit of 100.

call stack overflow

Definitions can be nested only to 20 deep.

EOF in string

An end-of-file was encountered before a bracketed string was terminated.

evaluation stack overflow

Too many characters are used for the name, definition, and arguments of one macro. 2500 characters are allowed.

unexpected EOF

An end-of-file was reached before a macro definition was terminated.

filename: can't open

The named file can not be opened.

filename: can't include

The indicated file cannot be included with the built-in macro **includ**.

includes nested too deeply

Files included with the built-in macro **includ** can be nested only up to 128 deep.

expression: invalid infix expression

There is a syntax error in the indicated infix expression as passed to the built-in macro **expr**.

too many characters pushed back

A macro expansion is too large to be rescanned. A macro definition may contain up to 2500 characters.

name: too many definitions

The table space for macro definitions has been exhausted; this occurred upon the definition of the named macro.

token too long

A name or symbol in the input was longer than the token buffer. Each token may be up to 200 characters long.

NAME

mbd – dump usage info on tcp buffer pool

SYNOPSIS

/etc/mbd [**-f**] [**-k**]

DESCRIPTION

The **mbd** command dumps usage information about the tcp memory buffer pools. Usage statistics on tcp memory buffers may be obtained by using the **-m** option of the **netstat** command; **mbd** is intended for analyzing cases where buffers are being lost. It scans the entire buffer pool, finding all the chains of in-use buffers; it then prints each chain of buffers. This information may help you in discovering reasons why buffers are being lost.

OPTIONS

- f** Dump the free pools as well as the chains of in-use buffers. This produces a lot of output.
- k** Don't try to lock the mutex on the buffer pools before doing the dump. This is useful mostly when the **tcpd** has crashed with the buffer pool mutex locked.

EXAMPLES

A dump of the buffer pools of a basically idle tcp might look like this:

\$ /etc/mbd

```
Offset 0x000035cc  size 1520 type 1 off 24 len 1512 refcnt 1 pool 1
Offset 0x000041cc  size 1520 type 1 off 24 len 1512 refcnt 1 pool 1
Offset 0x00003bcc  size 1520 type 1 off 24 len 1512 refcnt 1 pool 1
Offset 0x0000a7cc  size 1520 type 1 off 24 len 1512 refcnt 1 pool 1
Offset 0x00004dcc  size 1520 type 1 off 24 len 1512 refcnt 1 pool 1
Offset 0x00007dcc  size 1520 type 1 off 24 len 1512 refcnt 1 pool 1
```

Here there are 6 large (1520-byte) buffers in use, all on a single chain.

NAME

mkapr – make an Apollo product report

SYNOPSIS

mkapr [-v]

DESCRIPTION

The **mkapr** command creates a product report. This command replaces the **crucr** (create a user change request form) utility available in prior software releases.

Output from **mkapr** may be in either (or both) of two forms:

1. Printed, human-readable copy; or
2. Encoded, transmittable form.

Printed product reports should be sent to:

APR Administrator/Customer Services
M/S CHG 01 CS
Apollo Computer Inc.
330 Billerica Road
Chelmsford, MA 01824

Encoded product reports may be sent to Apollo Customer Services via the UUCP network. The network address is: **apollo!apr_cs_admin**

Recommended paths to Apollo are via **attunix**, **mit-eddie**, or **decwrld!decvax** (these paths may change). Customer Services will acknowledge all product reports received. Do not assume your product report has been received until you receive a reply. Security-conscious sites should not send confidential material. Voluminous submissions should be sent via magnetic media.

OPTIONS

mkapr supports only one option, **-v**. This will assert verbose mode; any system services called by **mkapr** will be allowed to send output to the standard output and/or standard error devices. Normal mode operation is for **mkapr** to invoke the system services silently.

SERVICES SUPPORTED

In addition to creating Apollo Problem Reports online, **mkapr** will make available viewing, editing, printing and mailing services if they exist (and **mkapr** can find them). The mailing service known to **mkapr** is:

UNIX environment - **sendmail**

The print services known to **mkapr** are:

Aegis environment - **prf**
 BSD environment - **lpr**
 SysV environment - **lp**

If a desired service is not available to **mkapr**, a product report (print or send) file will be saved in the current directory for printing or sending at a later time.

DIALOG INTERFACE

mkapr will make use of the DIALOG graphic interface environment of the Apollo Domain system whenever possible. This interface is designed for ease of learning and use.

COMMAND DRIVEN INTERFACE

If the display environment you are using does not support the graphic interface, you will see the following prompt:

```
mkapr>
```

Entering the command 'help' will display the available commands. Here is the list of commands for reference:

Command	Description
help [mkapr]	List Commands. To display the help file, use the mkapr option.
change	Change APR Information Fields.
edit	Edit the detailed Problem Description.
view	View the current APR.
print	Print the current APR.
send	Send the current APR.
exit	Save current customer information changes (if any) and exit.
cancel	Exit without saving customer information changes.

You need only enter as much of any command as is necessary to uniquely identify it. For example, you need only type **ch** for the change command.

Detailed descriptions of commands

change Allow user to provide the necessary information prior to submitting an APR. There are 2 kinds of input here. First, information that is extracted from the system the user is on. Second, information that the user must input. Most field defaults (including system extractable data) will be overridable by the user. The date field is the only non-

overridable field. A file exists between sessions which currently stores customer contact, name, address, and telephone information. This file is created upon the first invocation of the `mkapr` tool, is stored in the current working directory and is called `.aprint`. Upon subsequent invocations of the `mkapr` tool, the customer information is used as the default for these fields.

Within the change command, the prompt becomes `mkapr..change>`. Current input is then displayed by field. The user is asked to enter the field # to change, then asked to enter the changed value (entering `<RETURN>` effectively will abort the current change field # request leaving the field unchanged). The cycle is then repeated. Replying 'h' or 'help' at this point will display the following help message for the change command:

Change Command	Description
<code>help [mkapr]</code>	List commands. To display the help file, use the <code>mkapr</code> option.
<code>display fields</code>	Display all fields and their respective values.
<code>change field n</code>	Request to change the value of field # <i>n</i> . Pressing the RETURN key at the prompt <p style="text-align: center;">enter new value ==></p> will leave the value unchanged.
<code>exit</code>	Exit the change command.
<code>edit</code>	An appropriate editor will be invoked according to available system services. The user should enter a detailed problem description and save and exit the editor in the appropriate manner. You will then be returned to the <code>mkapr></code> prompt.
<code>view</code>	The current <code>mkapr</code> information will be displayed to the user in an appropriate manner according to available system services.
<code>print</code>	The current <code>mkapr</code> information will be printed to the default printer according to available system services.
<code>send</code>	The current <code>mkapr</code> information will be sent to Apollo Computer in an appropriate manner according to available system services.
<code>exit</code>	If any changes to customer information occurred during this session, save all customer information to the non-system
<code>cancel</code>	Exit <code>mkapr</code> . Do not save changes to customer information from this session.

INITIAL FIELD VALUES

The fields of an Apollo Problem Report that are collectively known as customer information Fields are initialized from a file read when `mkapr` starts up. These fields contain such information as the name of the customer contact, the name (company name) of the customer, and the customer's address and telephone number. The initialization file has the name `.aprinit` and the `mkapr` program will search for it. The search order for the initialization file is:

1. Look in the current working directory
2. Look in the home directory as given by the shell variable `HOME`
3. Look in the system directory `/etc/apr`

It is not an error for no initialization file to exist; `mkapr` will leave the customer information fields blank. The fields can be edited and the initialization file will be updated when `mkapr` exits.

The file `/etc/apr/.aprinit` is a special case; `mkapr` will not write to this file. The system administrator (or other privileged account) must create the directory `/etc/apr` with appropriate access permissions, then run `mkapr` to create a local copy of the file `.aprinit` and copy or move the file to the directory.

The initialization file is an ASCII text file that may be created and modified using any of the text editors available to you. The body of the `.aprinit` file created by `mkapr` is reproduced here:

```
# Comment lines begin with '#'
# Non-comment lines have the following form:
# FIELD_NAME : FIELD_VALUE : IGNORED
# The field name must not be changed.
# The ':' character delimits fields.
# The field value may be changed; it must not contain ':'.
# unless the field value is quoted by either ' ' or " " pairs.
# Anything after the second ':' is thrown away.
#
customer_contact : A. Random User : 14
customer_name : Apollo Computer, Inc. : 21
customer_addr1 : CHF 02 RD : 9
customer_addr2 : 330 Billerica Road : 18
customer_addr3 : Chelmsford, MA 01824 : 21
customer_addr4 : USA : 3
customer_phone : 1-508-256-6600 x7739 : 20
mail_path : 'apollo!apr_cs_admin.' : 22
#
```

NOTES

Since **mkapr** assumes that the site mail facility (probably **sendmail**) knows how to get from your site to Apollo, you must edit the **mail_path** field value in **.aprint** to give **mkapr** the correct path. Be sure that your mail facility is setup correctly. See your site administrator for help.

Run **/usr/ucb/newaliases** at least once before attempting to use **mkapr**'s send function.

Offsite mailing may not be allowed by your site. If so, you must make other arrangements to get mail to Apollo. See your site administrator for help.

FILES

/usr/apollo/bin/mkapr	The executable object
/usr/man/cat1/mkapr.1	This manual page (UNIX)
/sys/help/mkapr.hlp	This help file (AEGIS)
	Initial field values (search order):
.aprint	(1st) (updated)
\$HOME/.aprint	(2nd) (updated)
/etc/apr/.aprint	(last) (read only)
	Temporary files:
/tmp/apr.*	Temporary files:
apr.*.v	Product report view file
apr.*.p	Product report print file
apr.*.s	Product report send file
apr.*.c	Product report send command file
apr.*.e	Problem description edit file

NAME

mkcon – set console device

SYNOPSIS

/etc/mkcon [-p] [-d *dev*] [-c *cmd*] [-n]

DESCRIPTION

If no arguments are specified with this command, it makes the current controlling terminal into a console and starts up a shell. The shell type is determined by the shell environment variable. When the shell exits, the console output is redirected to *'node_data/system_logs/console'*.

OPTIONS

- p Create a new DM pad for the console in place of the controlling terminal device.
- d *dev* Make *dev* replace the controlling terminal device as console.
- c *cmd* Execute *cmd* instead of \$SHELL.
- n Do not run a shell.

EXAMPLE

```
$/etc/mkcon -d /dev/display -n
```

This causes console output to appear in a DM window, with a new window each time */dev/console* is opened.

NAME

mkdev – shell script to make devices

SYNOPSIS

mkdev *device_directory* [-**d** *devno_file*] [*all* | *console* | *tty* | *null* | *sio* | *pad* | *pty* | *dsk* | *mt* | *global_devices* | *crp*]

DESCRIPTION

mkdev creates devices. *device_directory* is usually /dev; **-d devno_file** can be used to specify a device number/manager mapping file to use in place of *'node_data/device_numbers'*.

If no additional arguments are specified, then all devices which have not been created will be. **mkdev** creates a file called *.mkdev* in the device directory to record for future instances of itself what work has to be done, (actually just the version of the last **mkdev** to run to completion).

If any arguments are specied (or *all*) then these devices will be deleted and recreated.

DIAGNOSTICS

Should be self-explanatory.

FILES

'node_data/device_numbers' Default device number mapping file

NAME

mtvol – mount a logical volume

SYNOPSIS

mtvol *disk_type*[*unit*] [*log_vol_number*] [*pathname*] [*options*]

DESCRIPTION

A logical volume is a named storage area on a disk. **mtvol** mounts a logical volume, making the files and directories it contains accessible. Up to eight volumes (both physical and logical) may be mounted on a node at any time. No more than five of the eight volumes may be logical.

Before a new physical volume can be mounted for the first time, you must initialize it. See the **invol** (**initialize_volume**) command description for details.

ARGUMENTS

disk_type (required)

Specify the type of disk on which the volume being mounted resides. Valid disk types are: w (winchester), s (storage module), or f (floppy).

unit (optional)

Specify disk unit number (0 or 1). If you use this argument, the unit number must follow the *disk_type* ID immediately: with no blanks in between. For example, "S1" denotes storage module unit 1.

Default if omitted: 0

log_vol_number (optional)

Specify the disk volume number. This is the same number that you assigned when you formatted the disk using **invol**. The first logical volume is numbered 1; the second 2; and so forth.

Default if omitted: 1

pathname (optional)

Specify the name of the volume entry directory. This is the logical volume's top-level directory. Specify this *pathname* only if the entry directory is not already cataloged in the naming tree. If the *pathname* you choose already exists, an error results.

Logical volume entry directories may appear anywhere in the naming tree, with one exception: if a logical volume entry directory is also the node's entry (top-level) directory, it must appear just below the network root directory (/).

If you omit the pathname argument, **mtvol** assumes that the entry directory already exists, and searches the naming tree for it. If it finds the entry directory, **mtvol** mounts the volume and prints the full entry directory pathname.

If **mtvol** does not find the entry directory, it prints an error message, and does not mount the volume. The search may fail for any of the following reasons:

- The entry directory has never been cataloged.
- The entry directory was uncataloged when the volume was last dismounted.
- The entry directory pathname exists on another node, for which directory information is currently unavailable.

An unsuccessful search does not mean that you cannot mount the volume. It simply means that the volume entry directory pathname does not exist on your node. To mount the volume, issue the **mtvol** command and supply an entry directory pathname.

Even if **mtvol** finds the entry directory pathname, the mount may fail if the volume is corrupt and needs salvaging. In this case, **mtvol** asks for permission to mount the volume. You should usually respond "no" to this request, then run the volume salvaging routine **salvol**. Once the volume has been salvaged, you may try to mount it again. If you mount a corrupt volume without salvaging it first, damage to files in that volume may result.

Default if omitted: (see above)

OPTIONS

- f** Force. Mount the volume whether or not it needs salvaging, and do not ask for permission.
- nq** No query. Suppress query if a volume needs salvaging. Instead, mount the volume only if it does not need salvaging.
- pr** Protect. Mount the volume with write protection. Any attempts to write on the volume will fail.

CAUTION

Before removing a floppy disk volume mounted with **mtvol**, you must use **dmtvol** to dismount it. Failure to dismount the volume could result in lost or corrupt information.

EXAMPLES

```
$ mtvol f /masterfloppy
Volume mounted, entry directory is /masterfloppy
$ dmtvol f
$ mtvol f
```

This command sequence mounts the floppy and makes a new entry directory, then dismount the floppy, and finally remounts it using the new entry directory.

SEE ALSO

More information is available. Type

help invol

NAME

mvf – move a file

SYNOPSIS

mvf *source* [*destination*] ... [*options*]

DESCRIPTION

mvf moves a file to a different location in the naming tree. Its effect is similar to

\$ cpf source destination \$ dlf source

Thus, it copies *source* to *destination*, and deletes *source*. **mvf** always retains the source ACL on objects moved.

ARGUMENTS

source (required)

Specify name of file to be moved. Wildcarding is permitted.

destination (optional)

Specify new file location. This pathname may be a derived name. If *destination* is a directory, the command moves the source file into that directory. Otherwise it creates the new file using the name specified.

Default if omitted: copy source to current working directory

Multiple source/target pairs and wildcarding are permitted.

OPTIONS

- p[airwise]** Instructs **mvf** to accept the list of tokens as consecutive pairs. This is necessary whether you are using wildcards or not.
- c** (default) Create the target file. If the target file already exists, an error results.
- r** Replace target file with source file. Use this option if the target file already exists. If the file does not exist, this option works like **-c**.
- du** Delete when unlocked. This option is useful with **-r**. If the object to be replaced is locked when **mvf** is invoked, the replace operation is performed when the object is unlocked.
- f** Force deletion of destination object if you have 'p' (protect) rights, even if you do not have 'd' (delete) rights.
- lf** List files moved.
- ldl** List files deleted by **-r** option.
- chn** Change the name of an existing destination file if required. This option modifies the meaning of **-c** and **-r**. If **-c** is specified, this option causes any existing object with the destination pathname to be renamed prior to the move. If **-r** is specified, the destination object is renamed if it is in use and cannot be deleted.

NOTES

If you use more than one pair of name tokens with this command, you must use the `-p` option. It instructs the command to accept the list of tokens as consecutive pairs. This is necessary whether you are using wildcards or not. In the past, this command has correctly paired off tokens without the prompting of a switch; now the `-p` switch must be used to achieve this result. The switch has been added to protect against inadvertent use in a shell, other than `/com/sh`, where wildcard expansion can be a problem.

EXAMPLES

```
$ mvf //anger/sam/letter //mary -r
```

Move the file `letter` from the directory `//anger/sam` to the directory `//mary` and replace the current file.

NAME

nd – set or display naming directory

SYNOPSIS

nd [*pathname*]

DESCRIPTION

nd sets or displays the name of the naming directory. The naming directory is provided so that you may use a tilde (~) as a shorthand feature in pathname specifications. It is also important since the system checks its **com** subdirectory (~/**com**) as a part of the default command search operation. The naming directory is set to the log-in home directory at login.

ARGUMENTS

pathname (optional) Specify directory name to be used as the naming directory. **nd** also accepts the command-line parser arguments "-" and "*".

Default if omitted: display the name of the current naming directory

EXAMPLES

Set naming directory to **/paul/links**.

§ **nd /paul/links**

After execution of this command, you can use a tilde (~) in place of **/paul/links** at the beginning of any pathname. Thus ~/sausage would be the same as **/paul/links/sausage**.

NAME

netmain – analyze network maintenance stats

SYNOPSIS

/etc/netmain [options]

DESCRIPTION

netmain is an interactive, menu-driven program that lets you control the `netmain_srvr`, the network maintenance server, and analyze the data that `netmain_srvr` produces. netmain provides detailed help from its menus.

OPTIONS

- `-w[help]` (default) Make sure the window is large enough to display command menus and interactive help.
- `-wc[md]` Set the window size smaller for command menus only. If you later decide that you want to see the helps, grow the window manually with `<GROW>`.
- `-nw` Do not change the window size.

EXAMPLES

1. Run netmain in a window large enough to display command menus and interactive help:

```
$ /etc/netmain
```

2. Run netmain in a window large enough (but no larger) to display the command menus:

```
$ /etc/netmain -wc
```

SEE ALSO

More information is available. Type

`help netmain_srvr` For details about gathering network error statistics

`help netmain_note` For information about adding notes to the network error log

NAME

netmain_chklog – clean up bad log files

SYNOPSIS

/etc/netmain_chklog [options] pathname...

DESCRIPTION

When the `netmain_srvr` program halts catastrophically (for instance, during a node reset), it can leave the log file it was writing in a corrupt, unusable state. `netmain_chklog` determines whether the log is corrupt and, optionally, deletes corrupt files.

If the pathname you specify points to some kind of file other than a `netmain` log file, that file is almost always ignored; it is almost never deleted as a corrupt log. On very rare occasions, another kind of file may look so much like a corrupt log that it might be deleted accidentally if you use both `-d` and the standard command option `-nq` (no query). Thus you should use `-d -nq` with extreme care.

pathname (required) Specify the files to be checked. Multiple names and wildcarding are permitted; separate names with blanks.

OPTIONS

<code>-d</code>	Delete corrupt log files.
<code>-nd</code> (default)	Do not delete anything.
<code>-l</code> (default)	Describe every file analyzed.
<code>-nl</code>	Describe only corrupt log files.

EXAMPLES

```
$ /etc/netmain_chklog 'node_data/net_log/*'
```

SEE ALSO

More information is available. Type

`help netmain_srvr` For details about `netmain`'s data collection server

NAME

netmain_note – place message in network error log

SYNOPSIS

/etc/netmain_note string [string ...]

DESCRIPTION

netmain_note sends a text string to **netmain_srvr**, the network maintenance server. The message is broadcast to all maintenance servers.

Typical topics of maintenance notes include known or explainable network failures, scheduled down-time, and node installations.

string (required) Specify message to be sent. You may use any string that is legal in a shell command. (Note that the shell takes special action on some keywords, such as 'if', unless you place them in quotation marks.) If there is more than one string, **netmain** builds the note by concatenating the arguments that are separated by spaces.

EXAMPLES

```
$ /etc/netmain_note 'Scheduled down time at 5 pm.'
```

```
$ /etc/netmain_note Cable disconnected at //sancho_panza
```

SEE ALSO

More information is available. Type

help netmain For instructions for controlling **netmain_srvr** after it starts, and for analyzing the data it collects

help netmain_srvr For details about **netmain**'s data collection server

NAME

netmain_srvr – collect network error stats

SYNOPSIS

/etc/netmain_srvr [options] [*pathname*]

DESCRIPTION

netmain_srvr collects and stores performance statistics for the Apollo token ring network. Use **netmain_srvr** to gather information; use the **netmain** program to display and analyze the information.

You can set parameters for **netmain_srvr** from the command line and from an options file. Once the server is running, you can change any parameter using the **netmain** program. To include parameters in an options file, specify the **-cmdf** option.

When you specify **-cmdf *pathname***, **netmain_srvr** reads the options listed in the options file first and then reads any other options on the **netmain_srvr** command line. If options specified in the file and on the command line differ, **netmain_srvr** uses the command line settings. For example, if the options file specifies a log file length as **-ll 1500**, and the command line specifies **-ll 3000**, **netmain_srvr** uses **-ll 3000**.

If a **netmain_srvr** does not start properly, a record of the failure appears in **'node_data/netmain_srvr.err_log'**.

OPTIONS

- a[ppend]** Append to an existing log file with this name, if one already exists; otherwise, create a log file with this name. This option is only valid when a log file *pathname* is specified with the **-l** option. Contrast this with the **-nappend** option.
- cmdf *pathname*** Accept options from an ASCII text file *pathname*. You may use this option only from the command line, not in the options file. There can only be one options file.
- l[og] [*pathname*] (default)** Create a log file. Optionally, specify a *pathname*, which is relative to the **'node_data/net_log'** directory. If either this option or the *pathname* is not specified, the log file name is derived from the current date: **'node_data/net_log/net_log.yy.mm.dd'**. The log file is stored on the disk of the node running **netmain_srvr**, and must remain there for **netmain_srvr** to write to it.
- ll *n* (default)** Set an upper limit on the length of the log file. The file size limit *n* is in 1024-byte units. The default value for *n* is 3000. You must use this option when you start the monitor and if you don't want to use the default length for the first log file, since you cannot change the name of a log file once it's open.

- na[ppend]** (default)
Create a new log file, over-writing any log with that name, if one exists. This option is only valid when a log file pathname is specified with the **-l** option. Contrast this with the **-append** option.
- nl[log]**
Do not write to a log file. `netmain_srvr` still runs probes and observers.
- ntopo[_init]** (default)
Override the **-topo_init** option, if **-topo_init** is specified in an options file. **-ntopo** is useful only on the command line.
- obs[erve]** *observer time ...*
Set the interval at which the named observer wakes up. Specify *time* as hh:mm:ss, hh:mm, or *never*, if you do not want the monitor to run the observer. Multiple observer/time pairs are permitted. See the default times listed below for each observer.
- re_obs[erve]** *observer time ...*
Set the "Recheck interval" — the interval that the observer waits before rechecking a node that has caused an alarm condition. By setting a recheck interval, you ensure that the observer only reports on a node once every *time* period. If the recheck interval is too short, the observer may produce many redundant alarms. Specify *time* as hh:mm:ss, or hh:mm. Multiple observer/time pairs are permitted. See the default recheck intervals listed below for each observer.
- s[ample]** *probe time ...*
Set the interval at which the named probe wakes up. Specify *time* as hh:mm:ss, hh:mm, or *never*, if you do not want the monitor to run the probe. Multiple probe/time pairs are permitted. See the default times listed below for each probe.
- sk[ip]** *probe distance ...*
Set the skip distance for the probe named. If the skip distance is *n*, the named probe samples approximately 1/*n* of the nodes every time it wakes up. Multiple probe/distance pairs are permitted. See the default skip distances listed below for each probe.
- topo[_init]** *pathname*
Initialize the monitor's total node list from a data file. The file may contain any number of node names or hexadecimal IDs, separated by spaces or on separate lines. If there is a "#" or "{" in any line, that character and all characters to the right of it are ignored (that is, "#" and "{" are comment markers).

EXAMPLES

1. Command: **cps /etc/netmain_srvr -ll 1500 -l tuesday_error_log**
2. Command: **cps /etc/netmain_srvr -s err_counts 0:15 hw_fail never**
3. Command: **cps /etc/netmain_srvr -cmdf /etc/start.net_srvr -ll 3000**

The file `/etc/start.net_srvr` might contain these lines:

```
-l -ll 1500
-sample err_counts 0:01:00 -skip err_counts 30
-sample topology 0:20:00
-sample disk_errs 0:01:00 -skip disk_errs 30
-sample time_skew never
-observe modem_errs 0:10:00
```

NAME

netstat – display network statistics

SYNOPSIS

netstat [*options*]

DESCRIPTION

netstat writes a summary of network and hard disk activity to standard output.

OPTIONS

If no options are specified, **netstat** returns a brief summary of network usage information for the current node.

- l** Long report—provides more information than the summary.
- config** Configuration report—displays only node-specific hardware information: CPU type, display type, etc.
- n *node_spec* ...** Provide information on specified node(s). See **help node_spec** for details about node specification syntax. Multiple *node_spec* strings are permitted; separate them with blanks.
- a** Report on all nodes in the network.
- r [*n*]** Repeat the **netstat** command every *n* seconds until halted by CTRL/Q. Only counts that have changed at each iteration are displayed, and the values represent the amount of change rather than absolute values. The default value for *n* is 10 seconds.
- s [*n*]** Send *n* test messages to every node being listed (except the current node) before every repeat of the display. If this option is specified, **-r** must also be specified. This option provides a minimum amount of network activity during the wait time between **netstat** repeats. The default value for *n* is 100 messages.
- save *pathname*** Save all statistics in the file named *pathname*.
- since *pathname*** Display counts that have changed since statistics were saved in *pathname*.

EXAMPLES

```
$ /etc/netstat
```

The node ID of this node is 1FB.

```
**** Node 1FB **** //diskless_0001Fb diskless to //anger
```

```
Up since 1988/02/01 at 8:17:06 Up for 1 day 2 hours 58 mins 4 secs
Net I/O: total= 94625 rcvs = 66912 xmits = 27713
Winchester I/O: total= 0 reads= 0 writes= 0 (NOTE 1)
```

System configured with 1.0 mb of memory.

\$ /etc/netstat -l

The node ID of this node is 1FB.

**** Node 1FB **** //diskless_\$0001Fb diskless to //anger

Up since 1988/02/01 at 8:17:06 Up for 1 day 2 hours 58 mins 52 sec:
Net I/O: total= 94756 rcvs = 67010 xmits = 27746

10436 page-in requests issued.
6473 page-out requests issued.
41134 page-in requests serviced.
12139 page-out requests serviced.

Detected concurrency violations -- read: 0 write: 0

Xmit count	27746	Rcv eor	0
NACKs	272	Rcv crc	767
WACKs	1639	Rcv timeout	0
Token inserted	65	Rcv buserr	0
Xmit overrun	0	Rcv overrun	0
Xmit Ack par	3	Rcv xmit-err	3042
Xmit Bus error	0	Rcv Modem err	0
Xmit timeout	90	Rcv Pkt error	45
Xmit Modem err	0	Rcv hdr chksum	0
Xmit Pkt error	377	Rcv Ack par	10

Delay switched OUT.

Winchester I/O: total= 0 reads= 0 writes= 0 {NOTE 1}

Not ready	0	Contrlr busy	0
Seek error	0	Equip check	0
Drive time out	0	Overrun	0
CRC error percentage: 0.00%			

Last ring hardware failure detected by node 241 {NOTE 2}
on 1988/02/02 at 10:05

System configured with 1.0 mb of memory.
A total of 0 ECC errors were detected.

Notes on Examples

1. Node 1FB is running diskless, hence the absence of Winchester disk I/O activity.
2. At 10:05 A.M. on Feb. 2, 1988, the network cable was disturbed immediately upstream of node 241. This information, coupled with the network topology available from `lnode` can help you pinpoint a hardware malfunction.

SEE ALSO

More information is available. Type

`help rtstat` For information about displaying internet router statistics

NAME

netsvc – set or display network services

SYNOPSIS

/etc/netsvc [*options*]

DESCRIPTION

netsvc sets or displays the network services that this node will perform. All changes take place immediately.

OPTIONS

If no options are specified, netsvc displays the network services allowed for this node.

- n** None. Disable all network services and physically disconnect this node from the network.
- l** Local. Allow only network requests originating at this node.
- r** Remote. Allow only network requests originating at other nodes.
- a (default)** All. Allow both locally and remotely initiated network requests. (The size of the remote paging pool is not changed.)
- s[ervers] [*n*]** Servers. Set the number of network servers to run on this node. At system startup, the number of network servers is 1. If this node is a network partner for diskless nodes or has several remote file users, their performance can be improved by increasing the number of servers. If *n* is not specified, the maximum number of servers (3) is used.
- p [*n*]** Pool. Set local memory pool size. Network page requests originating at remote nodes may not use more than *n* pages of the local node's memory. If *n* is not specified, all the local node's memory is eligible for remote page requests.
- net [*net_id*]** Network ID. Set or display network ID. Use this option to change or examine the ID of the network to which the node is attached. It affects only the node at which you type the command, not the rest of the network. Specifying a hexadecimal network ID changes your node's network ID. Using **-net** with no argument forces netsvc to display your network ID even if it is set to 0.

This option is useful only when there are no internet routers active on the node's network. Routers give the network ID to nonrouting nodes every 30 seconds, and may override the network ID you specify with this option.

NOTE

If the network ID you set with `-net` differs from the network ID used by other nodes on your network, your node may not be able to communicate with those other nodes.

Be careful when revoking network access with `-n` or `-l`. Remote file users may have problems, and writable files may be damaged. If your node was the network partner for a diskless node, that node will crash when your node leaves the network.

Use the `-s` option carefully. Although you can increase the number of servers, you cannot decrease it. The only way to return to a smaller number of servers is to reboot the node. Also note that increasing the number of server processes decreases the number of user processes allowed.

EXAMPLES

```
$ /etc/netsvc
Network operations allowed: ALL
Number of network servers: 1
Remotely initiated paging pool limit: NONE
Network ID: 437A9
$
```

SEE ALSO

More information is available. Type

`help rtsvc` For information about controlling a node's internet routing service

NAME

next – return to the top of a loop

SYNOPSIS

next

DESCRIPTION

next interrupts the flow of control in a shell loop construct (**for**, **select**, and **while**). When **next** is encountered in a **for** or **while** loop, control passes back to the top of the loop (see **EXAMPLES** below). When **next** is encountered in a **select** loop, control passes to the next case clause. This is useful when you have specified **select oneof** but want to test multiple things under certain circumstances.

You may terminate the flow of control in a loop by using the **exit** command. See the **exit** command description for more information.

The **next** command requires no arguments or options.

EXAMPLES

Consider the following section from a shell script:

```
n := 0
while ((^n < 10))
do   read -type integer n
     if ((^n < 10)) then next endif
     args ^n
enddo
```

As long as the **read** command reads integers into variable *n* that are less than 10, the **next** command executes and causes the script to return to the top of the **while** loop. When the value of *n* is greater than or equal to 10, the script prints the number then leaves the **while** loop and continues execution.

For more information on variables, refer to the manual, *Using Your Aegis Environment*.

NEXT

Aegis

NEXT

SEE ALSO

More information is available. Type

help exit For information on **exit**

help for For information on **for** loops

help select For information on **select** loops

help while For information on **while** loops

NAME

nor.dan_to_iso - convert files to ISO format

SYNOPSIS

nor.dan_to_iso *input_file output_file*

DESCRIPTION

These utilities convert files written with the overloaded 7-bit national fonts to the International Standards Organization (ISO) 8-bit format. The overloaded fonts include any with a specific language suffix (for example, *f7x13.french*, or *din_f7x11.german*). If you created and/or edited a file using one of the national fonts, that file is a candidate for conversion.

You are not required to convert files, but probably will want to because

1. The overloaded fonts replace certain ASCII characters with national ones, have that subset of ASCII characters and the national characters in one file. The 8-bit fonts available as of SR10 include all the ASCII characters and the national characters.
2. The 8-bit fonts also include a wider range of national characters, so you can enter any character in any western European language. This was not always possible with the overloaded fonts. For example, there was not enough space in the overloaded font to include all the French characters, but they all exist in the 8-bit fonts.

There are two parameters to the conversion utilities. You must provide the name of the file you want to convert (*input_file*) and your *output_file*. If *output_file* already exists, the utilities abort.

The default location for the utilities is */usr/apollo/bin*.

FILES

<i>/usr/apollo/bin/french_to_iso</i>	Converts overloaded French to ISO format
<i>/usr/apollo/bin/german_to_iso</i>	Converts overloaded German to ISO format
<i>/usr/apollo/bin/nor.dan_to_iso</i>	Converts overloaded Norwegian/Danish to ISO format
<i>/usr/apollo/bin/swedish_to_iso</i>	Converts overloaded Swedish/Finnish to ISO format
<i>/usr/apollo/bin/swiss_to_iso</i>	Converts overloaded Swiss to ISO format
<i>/usr/apollo/bin/uk_to_iso</i>	Converts overloaded U.K. English to ISO format

DIAGNOSTICS

All messages are generally self-explanatory.

NAME

not – negate a Boolean value

SYNOPSIS

not *command*

DESCRIPTION

not takes the Boolean value returned by a command or expression and negates it. This is useful primarily with the program control structures (**if**, **while**, etc.) used in shell scripts.

ARGUMENTS

command (required) Specify a command or expression that returns a Boolean value.

EXAMPLES

Assume the following lines appear inside shell scripts.

```
#
# Loop as long as no error file exists.
#
while not existf error_file
do args "No error file yet ..."
enddo
# End of script

#
# Verify user response.
#
eon
read -p "Type the pathname of the file to be deleted: " name
read -p "Are you sure you want to delete ^name?" verification
if ((^verification = "yes")) then
    delete := true
else
    delete := false
endif

if (( not ^delete )) then
    args "^name not deleted."
else
    dlf ^name -l
endif
# End of script
```

NAME

obj2coff – convert OBJ format modules to COFF format modules

SYNOPSIS

obj2coff *objmodule* *coffmodule*

DESCRIPTION

The **obj2coff** command converts SR9.5 or later object format modules to SR10 COFF format modules. Either individual modules, or complete bound programs may be converted.

This command cannot be used to convert object module libraries, see **lbr2ar**(1) for that purpose.

BUGS

If **obj2coff** encounters an object module stamped with an SR8 systype (*sys3*, *bsd4.1*, or any SR8), it converts it to COFF but does not change the systype, and issues a warning:

```
module is stamped with obsolete systype 'systype_name'
```

Some UNIX system calls may behave differently between *sys3* and *sys5*, or between *bsd4.1* and *bsd4.2*, so users are cautioned to examine their programs carefully for any effects caused by changes in system call semantics.

For object format files, streams 2 and 3 are used for error input and error output, respectively. No error input stream is automatically assigned for COFF format files; stream 2 is assigned to error output. Thus an object file which has been converted to COFF format may not work if it attempts to read error input. Moreover, if it writes to error output, the error "operation attempted on unopened stream" will occur.

SEE ALSO

More information is available. Type

lbr2ar For more information on converting lbr libraries to SR10 archive libraries

NAME

obty – set or display the type of an object

SYNOPSIS

/etc/obty (*[object_type]* *pathname...*)

DESCRIPTION

obty is intended for system-level debugging use only. Misuse of this command can cause objects to become inaccessible and programs to behave incorrectly.

pathname (required) Specify object whose type is to be set or displayed. Wildcarding of this *pathname* is permitted.

object_type (optional) Specify new type setting. *object_type* must be a known type; the **lty** command lists the types currently defined on a volume.

Executable files (output of compilers and binders) are **obj**, **coff** or **unstruct**. Most other binary files are **rec**.

Default if omitted: display current type of *pathname*

EXAMPLES

The sequence of the following commands is significant.

Display current object type:

```
$ /etc/obty testfile
"testfile" object type is nil.
```

Set type to **uasc**:

```
$ /etc/obty testfile uasc
```

Display new object type:

```
$ /etc/obty testfile
"testfile" object type is uasc.
```

NAME

os – convert ASCII to FORTRAN carriage control

SYNOPSIS

os [*pathname*...]

DESCRIPTION

os converts a file containing ASCII carriage control (for such things as form feeds and backspacing for underlining) into a file that can be printed on a line printer with FORTRAN carriage control. By default, output is written to standard output; redirect it into a file with the *>pathname* expression.

If you create a new file containing the overstruck text, **os** automatically sets the file's carriage control flag so that printers we supply interpret the file correctly. If you use **os** in a pipeline, however, the flag is not set (since output goes to standard output). In this case, you must use the **-ftn** option on the **prf** command for the file to be printed correctly. See examples 2 and 3 below.

ARGUMENTS

pathname (optional) Specify the file to be converted. Multiple pathnames are permitted, separated by blanks. However, all output is concatenated.

Default if omitted: read standard input

EXAMPLES

1. Convert the file **mary** and write to standard output.

```
$ os mary
$
```

2. Format the file **letter**, pipe output to **os**, and write the results into **letter.os**. This file is then printed on the default printer.

```
$ fmt letter | os >letter.overstruck
$ prf letter.os -npag
$
```

3. Format the file **letter** and pipe it directly to the line printer. Note the use of **-ftn** to ensure that proper carriage control is used.

```
$ fmt letter | os | prf -npag -ftn
$
```


4. Format letter and print it on a Spinwriter printer. Since Spinwriters use ASCII carriage control, `os` and the `-ftn` option on `prf` are not needed.

```
$ fmt letter | prf -npag -pr spin
$
```

NAME

pagf – paginate a file

SYNOPSIS

pagf [*options*] [*pathname...*]

DESCRIPTION

pagf paginates the named files to standard output. Each file is printed as a sequence of pages. Each page is 66 lines long by default, including a six-line header and three-line footer. The header includes the filename, the date and time, and the page number.

ARGUMENTS

pathname (optional) Specify file to be formatted. Multiple pathnames are permitted separated by blanks.

Default if omitted: read standard input

OPTIONS

-l n Set the page length to *n* lines. The default page length is 66 lines.

EXAMPLES

Paginate the file **mary** into pages 20 lines long and write them to **mary.short**.

\$ pagf -l 20 mary >mary.short

NAME

ppri – set or display process priority

SYNOPSIS

ppri [*process_name...*!–uid *uid_high.uid_low*] [*options*]

DESCRIPTION

The process priority is an integer ranging from 1 (low) to 16 (high). When the operating system decides which process to run next, it chooses the process that currently has the highest priority. As a process executes, its priority increases as it waits for events (such as keyboard input) and decreases as it computes for long periods without waiting. By default, the priority is bounded by the range 3 through 14 when a process is created. The **ppri** command lets you change these bounds to any other numbers in the range of 1 to 16.

All processes inherit the priority settings of their parent processes.

ARGUMENTS

process_name... (optional)

Specify name of process whose priority is to be set or displayed. Multiple names and wildcarding are permitted. If the process does not have a name, use the –uid option (below).

Default if omitted: use current process

OPTIONS

If no options are specified, the current priority bounds are displayed.

–lo *n* Set priority lower boundary. *n* must be in the range 1-16 inclusive. If this option is omitted, the lower boundary is set to 3.

–hi *n* Set priority upper boundary. *n* must be in the range 1-16 inclusive. If this option is omitted, the upper boundary is set to 14.

–u[*id*] *uid_high.uid_low*

Specify the UID of an unnamed process whose priority is to be set or displayed. The UID can also be separated by a space (*uid_high uid_low*).

EXAMPLES

1. Display defaults for current process

```
$ ppri
```

```
my_shell: minimum_priority = 3, maximum priority = 14
```

2. Restrict process_7 to low priorities

```
$ ppri process_7 -lo 1 -hi 4
```

3. Current process will always have priority 12

```
$ ppri -lo 12 -hi 12
```

NAME

prf – queue a file for printing by Domain/OS Aegis print spooler

SYNOPSIS

prf [*options*] *pathname*...

DESCRIPTION

The **prf** command queues a file for printing. The file must be an ASCII stream (that is, text) file, a graphics map file (GMF), or a GPR bitmap object. After successfully queuing a file, **prf** displays a message containing the full pathname of the file that you queued.

You can execute **prf** once for each file that you want to print (specifying all the necessary options every time), or you can enter **prf**'s interactive mode and hand files to the program continuously. See the examples for a sample interactive session.

Files queued by **prf** are physically printed by **prsvr**, the print server, running as a background task under control of **prmgr**, the print manager.

When you invoke **prf**, it first sets all options to their default states. Next, it looks for the print options file called `user_data/startup.prf` unless you invoke **prf** with the `-ndb` option. If **prf** locates the option file, it executes the options contained in the file to configure the current session. Finally, it processes all options on the command line.

pathname (optional) Specify the file to be printed. Multiple pathnames and pathname wildcarding are permitted.

Default if omitted: read standard input

OPTIONS

The following options can appear on the shell command line or in **prf** interactive mode. In addition, you can place one or more options in a **prf** option file so that they are executed automatically whenever you invoke **prf**.

Many of the options have default values that are specified in the **prsvr** configuration file established for each printer in the network by the system administrator. If you omit these options, your file is printed using the values specified in the **prsvr** configuration file. For example, omission of the `-banner` option could cause your file to be printed with a banner page if the **prsvr** configuration file specifies one.

If no options are specified, the file is printed using ASCII carriage control, with pagination enabled, on the default printer as established by the system administrator.

Options Applying to All File Types

- inter[active]** Enter interactive mode.
- sea[rch_dir] {on|off}** Search through all the directories of all the active processes on your node for the file(s) to be printed. This option is most useful in interactive mode, when the working directory of the **prf** process may be different from the working directory of the file to be printed.

The default is off.
- cop[ies] n** Print multiple copies of the file, where *n* is the requested number of copies. If **-cop[ies]** is specified, *n* is required. The default is one copy.
- pr[inter]name** Specify the name of the printer that should print the file. This option is useful only if more than one printer is in use on the network, or if a printer has been assigned a nonstandard name with the **printer_name** configuration directive in the **prsvr** command. If you omit this option, **prf** uses the default printer name, **p**. Note that **p** is also the default printer name used by the print server.
- s[ite] spool_node_name** Use this option only if you are queuing jobs to a pre-SR10 print server connected to a spool directory (**/sys/print**) that is different from the one specified by your node. By default, SR10 printers find the spool node for you.
- nc[opy]** Print the specified file from its location in the user-specified directory, bypassing **/sys/print/spooler**. If you select this option, **prf** defaults to the no-delete (**-nd**) option. If you specify the delete (**-d**) option, the file is deleted at the completion of the print request. If you use this option (with or without the delete option), do not open and alter the print file before the print job is completed.
- d[elete] (default)** Delete the print file at the completion of the print job.
- nd[elete]** Do not delete the print file when the print server is finished printing it. This becomes default if **-nc** is specified.
- user[username]** Specify the user name that appears on the banner page of the printed file. The alarm facility of **prf** also uses this name to determine who should be notified when printing is complete (see **-sig** below). This means that this name must be a valid log-in name (unless you don't care about sending an alarm).

The default is the current log-in name.

-sig[nal] {alarm|off} Request an alarm server signal when the file has finished printing.

The default is off.

-ban[ner] [on|off] Enable/disable banner page. If the banner setting in the prsrvr configuration file is off, no banner is printed.

The default is on.

-config[_file] [pathname]

Specify a file containing further prf options, one per line. Do not use prefixed hyphens (-) with the option names in the configuration file. If *pathname* is omitted, prf executes the prf option file `~/user_data/startup_prf`.

-ndb

Suppress processing of the prf option file.

-trans[parent] [on|off]

off specifies that the file being printed is passed directly to the printer driver routine with no processing by the print server. The default is on.

-filter[_chain] *string*

Specifies a filter string that will be used by the print server to process the job. This option overrides the default processing done by the print server. It is most often used to invoke filters that have been added to the print server. The format of the string is "filter1 | filter2", where filter1 and filter2 are composed of strings of the form "type1\$type2" and "type2\$type3". Note that the output type of filter *n* must equal the input type of filter *n+1*.

-paper_size {a|b|legal|a3|a4|a5|b4|b5}

Select the paper size. You must specify one of the following size codes:

Code	Size in inches (mm)
a	8.50 x 11.00
b	11.00 x 17.00
legal	8.50 x 14.00
a3	11.69 x 16.54 (297mm x 420mm)
a4	8.27 x 11.69 (210mm x 297mm)
a5	5.38 x 8.27 (137mm x 210mm)
b4	9.84 x 13.90 (257mm x 364mm)
b5	5.93 x 9.89 (182mm x 257mm)

This option is available only for the Domain/Laser-26 and

APPLE LaserWriter* printers. Because `prf` assumes that the correct paper is in the printer's paper tray, you should check the paper tray before printing. The default paper size is specified in the `prsvr` configuration file.

Options Applying to Text Files Only

- `-margins [on|off]` Enable/disable margins generated by `prf`.
The default is on.
- `-top n` Top page margin, in inches. The default is a value specified in the `prsvr` configuration file.
- `-bot[tom] n` Bottom page margin, in inches. The default is a value specified in the `prsvr` configuration file.
- `-right n` Right margin, in inches. The default is 0 inches.
- `-left n` Left margin, in inches. The default is 0 inches.
- `-headers [on|off]` Enable/disable page headers and footers generated by `prf`. The default is on.
- `-head[_string] l-string/c-string/r-string`
Specify contents of left, center, and right components of the page header generated by `prf`. Components can be empty strings. The following special characters return the values indicated when they appear in the header strings:

Character	Return Value
@	= Escape character
#	= current Page number with 1 leading and 1 trailing space
%	= Current date
!	= Filename
&	= Filename's last time, date modified
*	= Insert a space in text string (literal spaces are not allowed)

Example: `-head !/Page#/%` produces a header with the filename in the left component, the string "Page" followed by the current page number in the center component, and the current date in the right component. The default header is a string specified in the `prsvr` configuration file.

- `-foot[_string] l-string/c-string/r-string`
Specify contents of page footers. The format is the same as for `-head` above. There is no default footer.

-ftn [on/off] Enable/disable FORTRAN carriage control. **-ftn on** causes the print server to use FORTRAN forms control even if the file does not have the FORTRAN carriage-control flag. Use of this option causes **prf** to interpret the first character of each line as a FORTRAN carriage control character (and not print it). This can be unfortunate if the file has ASCII carriage control, so be careful. **-ftn off** causes the print server to print the contents of column one rather than trying to interpret it as FORTRAN forms control. If this option is specified without **on** or **off**, **on** is assumed.

The default is off.

-wrap [on/off] Enable/disable automatic line wrapping. When enabled, **prf** wraps lines that exceed the right margin. When disabled, **prf** truncates lines that exceed the right margin. If this option is specified without **on** or **off**, **on** is assumed.

The default is off.

-col[umns] {1|2} Specify single-or double-column printing.

The default state is single column.

-lpi *n* Specify the line-spacing factor. *n* is an integer indicating the number of lines per inch.

The default is six lines per inch.

Options for Variable Font and Pitch

-pitch *n* Set the printer pitch (characters/inch). The following pitch settings are available on the printers indicated.

Printer	Pitch
Printronic*	10
Spinwriter*	12
IMAGEN*	8.5, 10, 12, 15, 17.1
GENICOM*	10, 12, 13.1, 16.7
Versatec*	12
LaserWriter*	1 to 100
Laser-26	1 to 100

-point *n* Set the point size for the font to be used. This is a real number that specifies size in points. A point equals 1/72 inch.

-weight {light|medium|bold}

Set the weight of the font to be used.

The default is **medium**.

-lq [on|off]

Specify that the document is to be printed in letter quality (**on**) or in draft (**off**) mode. With no argument, **on** is assumed when this option is invoked. If the option is not invoked, draft mode is the default.

Options Applying to Plot Files

-res[olution] *n*

Output plot resolution in dots per inch. If you specify a resolution not available on the particular printer, **prsvr** prints the file at the closest available resolution.

The default resolution is specified in the **prsvr** configuration file.

-white[_space] *n*

The amount of white space (in inches) to appear between multiple plots in one file.

The default is three inches.

-bw[_rev] [on|off]

Enable/disable black and white reversal for bitmaps. If no argument is specified, **on** is assumed. If the option is not invoked, black/white reversal is disabled.

-magn[ification] *n*

Specify bitmap magnification value. *n* is an integer in the range -1 to 16. The values have the following meanings:

-1 Selects auto-scaling to magnify the bitmap to fill the available page space.

0 Selects one-to-one scaling between the display and the printer for GMF bitmaps. (For GPR bitmaps, this translates to magnification 1.)

1-16 Selects the magnification indicated by value. Where 1 equals 1-to-1, 2 equals 2x, etc.

Default if omitted: *n* is 0

Options Applying to PostScript* Printers

The following options apply only for files sent to printers that contain the PostScript interpreter, such as the Domain/Laser-26 and APPLE LaserWriter* printers.

-post[script] [on|off]

Enable/disable PostScript interpretation. When enabled, the data is passed through the PostScript interpreter. When disabled, the data is printed as text, plot, or transparent data. If the option is not invoked, PostScript interpretation is disabled.

The default is on.

-orient[ation] {port[rait]|land[scape]}

Select the page orientation. **portrait** specifies that the text or x-axis of the bitmap is printed parallel to the short edge of the paper. **landscape** specifies that the text or x-axis of the bitmap is printed parallel to the long edge of the paper and perpendicular to the short leading edge.

The default is portrait.

Information Request Options

-check [-pr *printer_name*]

Checks for the existence of the specified printer. If the printer does not exist or is unavailable, an error message is returned.

-list_printers

Lists the names and status of all printers currently attached to the network.

-list_sites

Lists the names of all print managers currently in the network.

-sig_printer *printer_name* {-abort|-sus[pend]|cont[inue]}

Signals the printer to abort, suspend, or continue an active print job.

-pre10

Allows you to queue print requests to a pre-SR10 print server.

COMMANDS

Once **prf** has been invoked in interactive mode (see **-inter** above), it accepts the following interactive commands at the "prf> " prompt (in addition to the options already discussed).

p[rint] [*print_file_pathname*] [*options*]

Queue the specified file for printing.

q[uit]

Quit interactive mode and return to the shell.

sh[ell]	Create a shell command line. This command allows you to issue shell commands without leaving prf interactive mode. When you finish entering shell commands, type CTRL/Z . This returns you to prf interactive mode. Your previous prf option settings remain undisturbed by the intervening shell commands.
init[ialize]	Reset prf parameters to their default values.
r[ead] [printer]	List queue entries for the specified printer. If <i>printer</i> is omitted, the contents of the queue (determined by the current setting of -pr) are listed.
wd [pathname]	Execute the shell command wd (<i>working_directory</i>) to set or display the working directory.
get option	Display the value of the prf option specified. Use this command to show the settings of the various prf parameters.
can[cel] [job_id]	Cancel printing of the specified file at the current printer. Note that you must specify the job ID assigned by prmgr when the file is queued. Use the read command to display the names and job IDs of currently queued files. This command affects jobs in the print queue; it does not cancel a job being printed. To halt a job being printed, use -pr_sig with abort specified.

EXAMPLES

The following example, queues the file named **mary** for printing and forces FORTRAN carriage returns:

```
$ prf mary -ftn
"//node1/my_dir/mary" queued for printing.
$
```

The following example queues the file named **filex** to the printer queue on the node named **//tape**:

```
$ prf filex -s //tape
"//node1/my_dir/test_file.pas" queued for printing at site //tape
$
```

This example shows the types of commands that might appear in the default **prf** configuration file **~/user_data/startup.prf**:

```
pr ge
site //rye
foot %/my_file/&
```

The following example shows a sample interactive session:

```
$ prf -inter
prf> get pr
pr = p
prf> -pr cx
prf> get pr
pr = cx
prf> -pitch 20
prf> print test_file.pas
"//node1/my_dir/test_file.pas" queued for printing.
prf> q
$
```

This example illustrated running **prf** from an icon. To run **prf** interactively in a process devoted to it, insert the following command in the start-up file that you use to start the DM:

```
cp -i -c 'P' /com/prf -inter -n print_file
```

The above command creates a **prf** process and turns its window into an icon using the print icon character in (`/sys/dm/fonts/icons`). Issue the DM command **icon** to change the icon window into its full-size format.

NOTES

APPLE and LaserWriter are registered trademarks of Apple Computer, Inc.

Printronix is a trademark of Printronix, Inc.

Spinwriter is a registered trademark of NEC, Inc.

IMAGEN is a registered trademark of IMAGEN Corp.

GENICOM is a registered trademark of GENDICOM Corp.

Versatec is a trademark of Versatec, Inc.

PostScript is a registered trademark of Adobe Systems, Inc.

SEE ALSO

More information is available. Type

- | | |
|--------------------------|--|
| help prfd | For information about the menu-based prf command |
| help printer | For general information about printers supported in a Domain/OS network |
| help prsvr | For details about the print server |
| help prsvr/config | For an explanation of the prsvr configuration file and its directives, including their default values |
| help prmgr | For details about the print manager |

NAME

prmgr – start the print manager

SYNOPSIS

/sys/hardcopy/prmgr -cfg configuration_filename -n process_name

DESCRIPTION

Print managers coordinate user print requests generated by the **prf** command and control one or more print servers. Print servers are bound to the print managers in the print server configuration file.

When the print manager receives print requests, it checks to ensure that the requested printer exists. If it does not, the job is rejected. If the printer exists, the print manager notifies an appropriate print server. The print server processes the print job and informs the print manager of the ongoing status of the print job.

Starting the Print Manager

Print managers are started with the **prmgr** command. You can start the print manager from any node on which the **llbd** (NCS Local Location Broker daemon) is running on the print manager node. The print manager can be started as either a foreground or background process.

Print Manager Configuration File

As an option to the command, you supply the print manager configuration filename. The print manager configuration file defines the manager's spooling node and logical name and, if appropriate, invokes a print server that allows printing of pre-SR10 print jobs.

The print manager configuration file contains three items:

- The print manager logical name, which should identify the type and location of the printers serviced by the print manager
- The print manager's spooling node, a node that includes a **/sys/print** directory (not just a link to that directory)
- An option, **pre10q**, that allows the SR10 print environment to accept pre-SR10 print jobs

You must define a configuration file for each print manager. A sample configuration file defining a spooling node named **//flash** and a print manager named **r&d** is shown below:

```
spool_node = //flash
prmgr_name = r&d1
pre10q
```

ARGUMENTS

- cfg *configuration_filename*** The name of the print manager configuration file. **prmgr** looks for the configuration file in the current working directory or specified pathname.
- n *process_name*** The name assigned to the print manager process. We recommend that you use the logical name specified in the configuration file.

SEE ALSO

More information is available. Type

- help prf** For information about printing files
- help prfd** For information about the menu-based **prf** command
- help printer** For general information about printers supported in a Domain/OS network
- help prsvr** For information about the print server
- help prsvr/config** For information about the print server configuration file

NAME

probenet – probe network and display error statistics

SYNOPSIS

/etc/probenet [options]

DESCRIPTION

This command broadcasts packets to the diagnostic socket in all nodes, then requests error counts indicating the status the broadcast was received with. It compiles counts from every node in the topology list and reports them to standard output.

OPTIONS

Use one of the following three options to specify the list of nodes to display:

- a** (default) Probe all nodes responding to a */com/lcnode* command. If the network is completely corrupted so that messages cannot make a complete pass, use one of the other two options to specify precisely which nodes to test.
- t *pathname*** Probe the nodes listed in the topology file indicated. The file must contain one hexadecimal node ID per line. Any text following a space after the node ID is ignored. You can insert comment lines if they are prefixed with a "#" or "!".
- n *node_id* ...** Probe the node(s) specified by the indicated hexadecimal node ID(s). A good choice of nodes to test is a set evenly spaced around the network.

Use the following options to specify which test to run:

- s *n*** (default) Specify the total number of packets to be sent to each node. The default number of packets is 10. If 0 is specified for *n*, no test messages are sent, but statistics from each node are collected.
- r [*n*]** Repeat the **probenet** cycle every *n* seconds. If *n* is omitted, the cycle is repeated every 10 seconds. When you press <RETURN> at the input window, the send cycle is terminated immediately and the statistics are gathered and reported.

Use the following options to specify which packets are sent:

- d *data_file*** Specify that the packets are taken out of the specified data file instead of the standard built-in data pattern.
- len *n*** (default) Specify the length (in bytes) of the data portion of the test packet, in bytes. The default length is 1024 bytes.

Use the following options to control the level of detail in the statistics report.

- l** Print long (detailed) error counts if there were any errors (that is, at least one transmit error (**xmit errs**) or receive error (**rcv errs**)).
- err** Print header for each test, but statistics only for nodes which returned errors (**xmit and/or rcv errs**).

-mon fail_lim

Print header for each pass, but statistics only on passes whose total failure count equal or exceed the *fail_lim* value.

-sens threshold

Open a window pane and select some output lines to append to this pane. The nodes selected are those whose error count exceed a five-node running average error count by the specified threshold value. Also, all nodes with modem errors are appended to this pane. The use of this secondary output is to do some data reduction and pick the nodes at or near points of data corruption in the network. The window pane is also stored in a named pad file called *probenet.pane*.

EXAMPLES

1. `$/etc/probenet`

{Probe entire network once. No errors detected.}

There are 5 nodes in the test.

Broadcasting 10 1024-byte packets . 85/02/20 21:16:52 # failures = 0

Last Biph hardware failure detected by node 676 on 85/02/20 at 19:15

NODE	NAME	ATTEMPT	MODEM		BIPH	ESB	TOKENS=	0
			ERRS	ERRS				
584	*diskless	10	0	0	0	0	0	Self
21	OS	10	0	0	0	0	0	
AEF	BS	10	0	0	0	0	0	
4A	HUBRIS	10	0	0	0	0	0	
3536	*diskless	10	0	0	0	0	0	

2. \$ probenet -t node_list -s 14400 -r 3600 -d data_e3

{ Probes network and displays nodes specified in file "node_list". This node broadcasts 14400 packets in 3600 seconds, that is, four packets per second. The packet data comes out of file "data_e3".}

There are 5 nodes in the test.

Broadcasting 14400 1024-byte packets (page 0) over 3600 seconds.

85/02/20 21:58:19 # failures = 100

Last Biph hardware failure detected by node 506 on 85/02/20 at 21:50

NODE	NAME	MODEM			BIPH	ESB	TOKENS=	3
		ATTEMPT	ERRS	ERRS				
1967	GTX	14386	0	0	0	0	1	Self
15F5	SWI	14386	0	0	0	0	0	
2255	BIRDIE	14384	0	0	0	0	1	
3FD	FLASH	14386	3	3	3	0	0	
2B69	STANG	14385	3	0	0	0	1	

Broadcasting 14400 1024-byte packets (page 0) over 3600 seconds.

85/02/20 21:58:41 # failures = 100

Last Biph hardware failure detected by node 506 on 85/02/20 at 21:50

NODE	NAME	MODEM			BIPH	ESB	TOKENS=	3
		ATTEMPT	ERRS	ERRS				
1967	GTX	14383	0	0	0	0	1	Self
15F5	SWI	14383	0	0	0	0	0	
2255	BIRDIE	14381	0	0	0	0	1	
3FD	FLASH	14383	4	4	4	0	0	
2B69	STANG	14382	4	0	0	0	1	

{ Above example shows a problem between node 3FD and its predecessor in the network. }

SEE ALSO

More information is available. Type

help lcnode

For details about determining which nodes are currently connected to the network

NAME

prsvr – start the print server

SYNOPSIS

/sys/hardcopy/prsvr [config_file_name] [-n name]

DESCRIPTION

prsvr is the command that starts the print server process, which handles the processing of files submitted by the print manager for printing. Print servers determine the print parameters and send print requests to a selected printer. They are bound to the print managers in the print-server configuration file.

Print servers are started (typically as a background task) by executing the **prsvr** command. **-n** specifies the name of the printer configuration file that defines the printer and its print parameters. You must execute this command whenever you start or restart the node connected to the printer. To avoid losing print files already queued, do not execute the **prsvr** command at nodes without an attached printer.

ARGUMENTS

config_file_name The name of the print-server configuration file.

OPTIONS

-n name The name of the print-server process. The default is **print_server.printername**. *printername* is the device name specified in the print server configuration file.

SEE ALSO

More information is available. Type

help prf	For information about printing files
help prfd	For information about the menu-based prf command
help printer	For general information about printers supported in a Domain/OS network
help prsvr/config	For an explanation of the prsvr configuration file and its directives, including their default values
help prmgr	For details about the print manager

NAME

pst – list process internal state information

SYNOPSIS

pst [*options*]

DESCRIPTION

pst lists internal state information for all processes in the system by name or UID.

OPTIONS

- r[*repeat*] *n*** Repeat every *n* seconds. If you include this option, the first pass displays the total time elapsed since process creation. Subsequent passes display changes from the previous pass, as shown in the first example below.
- n[*ode*] *node_spec*** Specify remote node whose process statistics are to be listed. See help *node_spec* for details about node specification syntax.
- un** Display Domain/OS process IDs.
- pa[*ging*]** Display process-paging information. The paging data presented is private page faults, global page faults, disk-paging I/O, and network-paging I/O.
- c** Display only brief information on level 2 (user) processes. This output also suppresses the header lines and the processor time total. See Example 5 below.
- ty[*pe*]** Show whether each process is a user process (stops at logout), a server process (started via *-cps*), or an Aegis process (internal to the operating system).

EXAMPLES

\$ pst

Node: 4DC0

Time: Thursday, May 26, 1988 2:50:12 pm (EDT)

Processor Time (sec)	PRIORITY mn/cu/mx	Program Counter	State	Process Name
13561.199	-- -- --	-----	-----	<Null Process>
185.307	-- -- --	-----	-----	<Aegis Processes>
442.032	16/16/16	9D63AA	Wait	display_manager
5.014	3/14/14	9D63AA	Wait	server_process_manager
1.158	3/14/14	9D5EAA	Wait	mbx_helper
16.708	3/12/14	<active>	Ready	aegis_shell
3.554	3/10/14	9D63AA	Wait	mail
21.927	3/14/14	9D63AA	Wait	alarm_server

PST

Aegis

PST

560.526	3/ 9/14	9D63AA	Wait	sys5_bourne_again
5.943	3/14/14	9D63AA	Wait	bsd4.2_c_shell
37.433	3/14/14	9D619A	Wait	lp26
411.974	3/14/14	9D63AA	Wait	vt100_server
10.631	3/11/14	9D63AA	Wait	uid = 3c495808.50004dc0

 15263.411

\$ pst -n //brazil

Node: CBB9

Time: Thursday, May 26, 1988 2:50:18 pm (EDT)

Processor	PRIORITY	Program	State	Process Name
Time (sec)	mn/cu/mx	Counter		

1329590.507	-- -- --	-----	-----	<Null Process>
42761.139	-- -- --	-----	-----	<Aegis Processes>
29.647	16/16/16	322752C	Wait	init
28.578	3/14/14	3226F5C	Wait	mbx_helper
36.040	3/14/14	32273EE	Wait	server_process_manager

1372445.914

SEE ALSO

More information is available. Type

help dspst

For information on displaying process status in a graphic format

NAME

rbak – restore or index a magnetic media backup file

SYNOPSIS

```
rbak {-f fileno|-fid id} [-dev | m[unit] | f | ct]
      [-int|-index] [-sla|-nsla] [-anys] [-reo] [-pr pn]
      [-cr|-r|-ms|-md] [-force] [-du] [-l|-ld|-lf|-ll]
      [-reten|-nreten] [-rewind] [-dacl|-sacl]
      [-from filename] [-pdt] [-stdin] {{-all|pn}
      [-as disk_pn]}...
```

DESCRIPTION

rbak restores objects from the backup input media written by **wbak** (`write_backup`). The backup input media can be one of magnetic media, file or standard input.

Use **wbak** and **rbak** to back up disks and to transfer information between separate Domain installations. (Use the `rwmt` (`read_write_magtape`) command to transfer information to and from non-Domain installations.)

rbak operates in either index or interchange mode. To restore objects to disk, use interchange mode (`-int`). To list object names on standard output, without restoring any information to disk, use index mode (`-index`).

pathname (optional) Specify name of the object to be indexed or restored to disk. This may be a directory, file, or link. If the object is being restored, the new disk object has the same name. If you wish the disk file to be saved under a different name, use `-as` (below). Multiple pathnames are permitted; however, wildcarding is not supported.

Default if omitted: must use `-all`

OPTIONS

Backup File Identifiers

One of the following options is required.

- `-f file_no` Read the back up file with the file number specified. You assigned this number with **wbak**.
- `-f cur` Begin reading at current position on the back up medium.
- `-fid file_id` Read the back up filename specified. You assigned this name using **wbak**.
- `-int` (default) Select interchange mode. Backup files are restored to disk.
- `-index` Select index mode. Backup filenames are listed on standard output; no information is restored to disk.

Catalog Control

- all** Restore or index all the objects in the back up file specified. This option is required if you do not use the *pathname* argument to indicate a particular object to be indexed or restored.
- as *pathname1*** Restore the object specified and assign a different disk pathname *pathname1*. This option is valid only when used with the *pathname* argument on the *rbak* command line.
- cr (default)** Specify create mode. *rbak* does not restore objects if their names already exist on disk. It prints an error message if a name exists on both disk and backup media, and continues.
- r** Specify replace mode. *rbak* deletes the existing disk object, and replaces it with the object read from backup media.
- force** Force object deletion if you have owner rights, even if you don't have delete rights.
- du** Delete when unlocked. If the object to be deleted is locked when *rbak* is invoked, the delete operation is performed when the object is unlocked.
- ms** Specify merge-source mode. Similar to replace mode. If an object already exists on disk, *rbak* deletes the disk version and restores the backup media version (the source). However, if the object is a directory, *rbak* merges the back up media directory's contents with the disk directory.
- pr *pathname...*** Preserve specified objects on the disk. Multiple pathnames and wildcarding are permitted. If the objects exist on disk, they are not overwritten by backup media versions. This option must be used with the **-ms** option.
- md** Specify merge-destination mode. Similar to create mode. If an object already exists on disk (the destination) *rbak* does not restore the backup media version, and retains the disk version. However, if the object is a directory *rbak* merges the backup media directory's contents with the disk directory.

Label Control

- sla (default)** Display the backup media file label on standard output.
- nsla** Do not display the backup media file label.

Listing Control

You may include the **-l** option, or any combination of **-ld**, **-lf**, and **-ll**.

- l** Write all the file, directory, and link names to standard output.

- ld** Write all directory names to standard output.
- lf** Write all filenames to standard output.
- ll** Write all linknames to standard output.

Backup Device Control

- anys** Force rbak to accept any section of the backup file. When a backup file spans multiple backup media volumes, rbak normally begins with the backup media volume containing the backup file's first section, and proceeds to the backup media volume containing the second section, and so on. If you know which backup media volume contains the object you want to restore or index, use this option. This lets rbak start at any section of the backup file.
- reo** Force previous volume to be reopened, and suppress reading of backup media volume label. Use only when backup media has not been repositioned since the last wbak or rbak.
- dev *d[unit]*** Specify device type and unit number. *d* must be either *m* (for reel-to-reel magnetic tape, *ct* (for cartridge tape), or *f* (for floppy), depending on which drive is being used. *unit* is an integer (0-3). Both are required for reel-to-reel tapes (that is, **-dev m2**). A unit number is not required for floppy disks and cartridge tapes (that is, **-dev f**). If this option is omitted, rbak assumes device *m0*.

Note: Floppy disk support for this command is limited. In particular, error detection during reads and writes is poor. Do not use this command with floppy disks when the data being placed on the floppy disks is critical and unrecoverable.
- from *filename*** The backup input can be read from a file written by wbak using the **-to** option. *filename* specifies the pathname of the file.
- stdin** Specify the backup input media to be standard input. Used along with I/O redirection, this option is useful for reading files from foreign file systems.
- reten** Retension the cartridge tape (unwind to the end, then rewind). This can be helpful if you encounter cartridge tape reading errors. Retensioning requires about 1.5 minutes to complete.
- nreten (default)** Do not retension the cartridge tape.

-rewind Rewind the cartridge tape after reading or indexing. If this option is omitted, the cartridge tape is left positioned to the next tape file. This option is valid only for the cartridge tape; reel-to-reel tapes are rewound automatically when removed from the drive.

ACL Control

-dacl (default) Assign the destination directory's default ACL to the object being restored.

-sacl Retain the restored object's original ACL.

-pdt Preserve the object's original date-time modified and date-time used.

EXAMPLES

```
$ rbak -f 1 fred/soup
```

Read **fred/soup** in backup file 1 and restore it to disk. **fred/soup** may be a directory, file, or link.

```
$ rbak -f 1 fred/soup -as //node5/noodle
```

Restore **fred/soup** and place it in **noodle** on **node5**.

```
$ rbak -dev ct -rewind
```

Rewind the cartridge tape prior to removing it from the tape unit.

```
$ rbak src -from /fred/bck_out.file
```

Restore the directory **src** to disk. Read the backup input from the file **/fred/bck_out.file**, that should be written by **wbak** using the **-stdout** or **-from** option.

```
$ catf /fred/bck_out.file | rbak src -stdin
```

Restore the directory **src** to disk. Read the backup input from standard input. Note that the file **/fred/bck_out.file** should have been written by **wbak** using the **-stdout** or **-from** option.

SEE ALSO

More information is available. Type

help wbak	For information on creating a magnetic media backup file
help rwmt	For information on reading/writing foreign magtapes
help media	For information on removable media

NAME

rdym – set system ready message

SYNOPSIS

rdym {**-on** | **-off**}

DESCRIPTION

rdym enables or inhibits the output of a system ready message to standard output after execution of each shell command. The message lists the CPU time required to execute the command and the elapsed time since the last command. Both times are reported in seconds and decimal fractions of seconds. The message appears on a line following the echoed command line in the shell's process transcript pad.

Turning on the ready message interactively or in a shell script causes it to be printed after each command of the program is executed. If the ready message is not disabled at the end of the shell script, it remains in effect after the shell script exits.

Ready message printing is enabled and disabled for levels. The level number increases each time a shell script or the shell is invoked, and decreases when it exits. The times printed in the ready message reflect the CPU and real time used since the last message at the same level. Thus, for example, if the ready message is enabled in a shell script, after the last command of the program is finished, two ready messages are printed: one showing the time used by that command; and the other the time used by the whole shell script.

If the ready message is turned on by one level, it remains on when that level exits; however, if it is disabled by a level, it reverts to its previous state when that level exits.

By default, system ready messages are disabled at login.

OPTIONS

-on	Enable message.
-off	Disable message.

EXAMPLES

```
$ rdyd -on
cpu time: 750.685.  real time: 5914.532.
$ bldt
      *** Node 29C27.CBB9 ***   "//brazil"
Domain/OS kernel(8), revision 10.0, bl17.3 \
      February 9, 1988  8:12:37 am

$ rdyd -off
$
```

NAME

read – set variables equal to input values

SYNOPSIS

read [*options*] [**-type** *type* *var_name* ... | *variable_list*]

DESCRIPTION

The **read** command reads input values and sets a list of variables to those values. The values from the input line are parsed as separate tokens (they must be separated by spaces), and each variable in the list is assigned the value of a token.

Use the **-p** *prompt* argument to instruct **read** to issue a prompt. If you do not input values for all the variables names listed as part of the **read** command, **read** displays a **more>** prompt to request further input. By default, the type of each variable specified in the **read** command depends on the type of each input value. However, you can use the **-type** argument to specify the individual type(s) of the the variables.

ARGUMENTS

variable_list (optional)

Specify the names of the variables that receive the input values.

Default if omitted: must specify **-type**

OPTIONS

-type *type* *var_name*...

Specify the type of the input value(s) that can be assigned to the particular variable name(s). Multiple variable names are permitted, separated by blanks. Once you specify a type in a particular **read** command, **read** assigns that type to all subsequent variable names, until you change the type specification. Valid types are

Type	Value
str [ing]	Character strings
int [eger]	Integer numbers
bool [ean]	Boolean values
env [ironment]	Environment variables
any	Any type (the default)

If the type of the input value does not match the type specified for that variable name, **read** issues an error and asks you enter another input value. Use **-type any** to restore the shell to its default state. In this case, it determines the proper variable type automatically.

Specifying `-type env var_name` causes the variable to become an environment variable. Environment variables are of primary concern to Domain/OS users; please consult the Domain/OS documentation for details about their usage.

- `-p[rompt] prompt` Specify a particular prompt string to request the input values. Enclose the string in single quotes if it contains literal blanks.
- `-err[in]` Read input from error input instead of standard input. This option is useful for reading user input from the shell's input pad (where error input is normally directed) when the read command appears inside a pipeline, since standard input in that case is connected to the pipe.

EXAMPLES

Consider the following command line in a shell script:

```
read -p 'Enter model and class:' model class
```

In this example, `read` displays the prompt "Enter model and class:" in the process input window, and assigns the input values to the variables named "model" and "class", in that order.

The following section illustrates how the `-type` option works. (The numbers in parentheses refer to the different parts of the example.)

```
$ read -p '>' -type integer tens ones -type string number (1)
> 40 four
<non-integer 'four'; please reenter> > (2)
<more> > forty-four (3)
$
$ lvar
integer tens = 40
integer ones = 4
string number = forty-four
$
```

- Line 1. We define the prompt to be "> ", specify variables "tens" and "ones" of type "integer", and specify variable "number" of type "string". This means that the read command expects its input to be three variables of types integer, integer, and string, in that order.

Line 2 Shows the error message and prompt when we enter the non-integer value "four", read cannot assign this value to variable "ones", and issues the error message and prompt.

Line 3. read prompts for the third input value.

Line 4 The lvar command displays the type, name, and value of the variables.

Here is a final example.

```
$ date | chpat ',' | (read day month date year; readln time)
$ lvar
string time = 12:40:42 pm (EST)
integer year = 1988
string month = December
string day = Wednesday
integer date = 14
$
```

In this example, the output from the date command is piped to chpat, which removes the commas and then sends its output to read and readln where the proper variable assignments are made.

SEE ALSO

More information is available. Type TP 1.5i readc For information on assigning single-character strings to variables

readln For information on assigning whole-line strings to variables

export For more details about environment variables

NAME

readc – set variables equal to input characters

SYNOPSIS

readc [*options*] *variable_list*

DESCRIPTION

The **readc** command reads single characters as input, and sets a list of variables equal to those character values. **readc** parses each character from the input line as a separate token, and each variable in the list is assigned the value of a token. Use the **-p** '*prompt*' argument to instruct **readc** to issue a prompt.

The **readc** command considers all input to be type *string*.

ARGUMENTS

variable_list (required) Specify the names of the variables that receive the input values.

OPTIONS

-p[*prompt*] *prompt* Specify a prompt string to request the input values. Enclose the string in single quotation marks if it contains literal blanks.

-err[*in*] Read input from error input instead of standard input. This option is useful for reading user input from the shell's input pad (where error input is normally directed) when the **readc** command appears inside a pipeline, since standard input in that case is connected to the pipe.

EXAMPLES

Consider the following sequence of commands and input:

```
$ readc -p 'Do you want to continue? (y/n): ' ans
Do you want to continue? (y/n): y
$ lvar
string ans = y
```

In this example, **readc** displays the prompt "Do you want to continue? (y/n): " in the process input window, and assigns the value of the first input character ("y" in this case) to the variable named "ans".

For more information on shell variables, refer to *Using your Aegis Environment*.

READC

Aegis

READC

SEE ALSO

More information is available. Type

help read For information on assigning multicharacter strings to variables

help readln For information on assigning whole-line strings to variables

NAME

readln – set a variable equal to an input value

SYNOPSIS

readln [*options*] *variable_list*

DESCRIPTION

The **readln** command reads a line of input and sets a variable to that value. Use the **-p** '*prompt*' argument to instruct **readln** to issue a prompt. **readln** accepts multiple variable names.

The variable type is always a string.

Refer to the descriptions of the **read** and **readc** commands for related information.

Consider the following command line in a shell script:

```
readln -p "Enter total here: " total
```

In this example, **readln** displays the prompt "Enter total here: " in the process input window, and assigns the value of the input line to the variable named "total."

ARGUMENTS

variable_list (required)

Specify the name(s) of the variable(s) that receives the input value(s). If you specify more than one variable name (separated by blanks), **readln** assigns the values of input lines to the variables in the order that the variables were named.

OPTIONS

- p[*prompt*]** *prompt* Specify a prompt string to request the input value. Enclose the string in single quotation marks if it contains literal blanks.
- err[*in*]** Read input from error input instead of standard input. This option is useful for reading user input from the shell's input pad (where error input is normally directed) when the **readln** command appears inside a pipeline, since standard input in that case is connected to the pipe.

EXAMPLES

Consider the following command line in a shell script:

```
readln -p 'Enter total here: ' total
```

In this example, `readln` displays the prompt "Enter total here: " in the process input window, and assigns the value of the input line to the variable named "total."

SEE ALSO

More information is available. Type

help readc For information on assigning single-character strings to variables

help read For information on assigning multicharacter strings to variables

NAME

return – return from current shell level

SYNOPSIS

return [*options*]

DESCRIPTION

The **return** command causes the shell to return from its current level with the specified status severity. See the **abtsev** command description for details about status severity levels.

OPTIONS

Specify one of the following options to select the return-severity level.

-ok	Set level to ok.
-t[rue] (default)	Set level to true.
-f[alse]	Set level to false.
-w[arning]	Set level to warning.
-e[rror]	Set level to error.
-o[utinv]	Set level to output invalid.
-i[ntfatal]	Set level to internal fatal error.
-p[gmflt]	Set level to program fatal error.
-m[ax_severity]	Set level to maximum severity error.

EXAMPLES

The following lines are a portion of a shell script:

```
#
# Test to see if the second parameter passed to the script is valid
# If it is not, abort the script.
#
if eqs ^1 '-test' then
  if eqs ^2 '-b' then
    cpf ^3 temp.mss
  else
    args "(?) >>> ^2 <<< is not a valid parameter."
    return -P
  endif
else
  cpf ^1 temp.mss
endif
```

SEE ALSO

More information is available. Type

help abtsev

For details about abort-severity levels

NAME

revl – reverse each line in a file

SYNOPSIS

revl [*pathname* ...]

DESCRIPTION

revl copies the named files to standard output, reversing the order of the characters in every line.

ARGUMENTS

pathname (optional) Specify name of file containing lines to be reversed.

Default if omitted: read standard input

EXAMPLES

Reverse a line from standard input.

```
$ revl
```

```
This command produces interesting results.
```

```
.stluser gnitseretni secudorp dnammoc sihT
```

```
*** EOF ***
```

```
$
```

Sort the system dictionary by suffixes to produce a rhyming dictionary.

```
$ revl /sys/dict | srf | revl >rhyming_dict
```

```
$
```

NAME

rgy_admin – registry server administrative tool

SYNOPSIS

/etc/rgy_admin

DESCRIPTION

The **rgy_admin** tool administers registry servers. It can view or modify the registry replica list, reinitialize replicas, delete replicas, stop servers, and change the registry master site.

Note that **rgy_admin** cannot add, delete, or modify data entries contained in the registry database, such as names and accounts; use **edrgy** to perform these tasks. To create a registry replica or to restart a server, use **rgyd**, the registry daemon.

Once invoked, **rgy_admin** enters an interactive mode in which it accepts the commands described in the next section.

COMMANDS

Most **rgy_admin** commands operate on a default host. You use the **set** command to establish the default host, which is remembered until changed by another **set**. In the following command descriptions, we identify the default host as *default_host*. If a command operates on a host other than the default, we identify this host as *other_host*.

Several of the **rgy_admin** commands require you to set the default host to the master registry site.

The host name you supply as a *default_host* or *other_host* takes the form *family:host* or *host*. The only currently supported *family* is *dds*, the Domain protocol family. You can specify a host in this family by its entry directory or by its network address. For example, *dds://clara*, *//clara*, *dds:#1234.abcd*, and *#1234.abcd* are all acceptable host names.

become [**-master**] [**-slave**] [**-ro** | **-wr**]

The **-master** option causes the replica at *default_host* (which must be a slave replica) to become the master. This operation can cause updates to be lost; the **change_master** command is the preferred means of designating a different master replica.

The **-slave** option causes the replica at *default_host* (which must be the master replica) to become a slave. This operation can cause updates to be lost; the **change_master** command is the preferred means of designating a different master replica.

The **-ro** option makes the replica list read-only. The *default_host* must be set to the master registry site.

The **-wr** option makes the replica list writable. The *default_host* must be set to the master registry site.

change_master *-to other_host*

Change the master replica of the registry from *default_host* to *other_host*. The *default_host* must be set to the master registry site.

The current master server copies its database to the replica at *other_host*, becomes a slave, then tells the replica at *other_host* to become the master.

delrep *other_host* [*-force*]

Delete the registry replica at *other_host*. The *default_host* must be set to the master registry site.

The master server marks the replica at *other_host* as deleted and propagates the deletion to all other replicas on its list. When it has actually delivered the delete request to the replica at *other_host*, the master server removes that replica from its own replica list.

The *-force* option causes a more drastic delete. It deletes *other_host* from the replica list at the master registry, which then propagates the delete request to the replicas at the hosts that remain on its list. Since this operation never communicates with the deleted replica, you should use *-force* only when the replica has died irrecoverably. If you use *-force* while the replica at *other_host* is still running, you should then reset the deleted replica.

help List the *rgy_admin* commands and show their allowed abbreviations.

info Get status information about the replica at *default_host*.

initrep *other_host*

Reinitialize the registry server at *other_host*. The *default_host* must be set to the master registry site. The *other_host* must be a slave site.

The master registry copies its entire database (or that of another up-to-date replica) to the replica at *other_host*.

lrep [*-state*] [*-na*]

List the registry replica sites as stored in the replica list at *default_host*.

The *-state* option shows the current state and update time on each host.

The *-na* option shows the network address of each host.

monitor [*-r m*]

Periodically list the registry replica sites as stored in the replica list at *default_host* and show the current state and update time at each site.

The *-r* option causes the sites to be listed every *m* minutes. If you omit this option, the period is 15 minutes.

quit Quit the *rgy_admin* session.

reprep *other_host*

Replace the network address for *other_host* in the registry replica list. The *default_host* must be set to the master registry site.

The master replica propagates the new network address for *other_host* to all other registry replica lists. Use this command only if a replica site's network number changes.

reset *other_host*

Reset the registry replica at *other_host*. The registry server at *other_host* deletes its copy of the registry and stops running. This command does not delete *other_host* from any replica lists.

set [*-h host_name* | *-m*]

Set the default host. Subsequent commands that do not specify a host will go to this host.

The *-h* option specifies a replica to use as the default.

The *-m* option causes the master replica to be the default.

With no options, *set* locates a registry replica and sets it as the default.

site [*host_name*]

If *host_name* is specified, the command sets the default host.

If *host_name* is not specified, the command gets status information about *default_host*.

state *-in_maintenance* | *-not_in_maintenance*

Put the master registry server into maintenance state or take it out of maintenance state. The *default_host* must be set to the master registry site.

With the *-in_maintenance* flag, *state* causes the master registry server to save its database onto disk and refuse any updates.

With the *-not_in_maintenance* flag, *state* causes the master registry server to reload its database from disk, return to its normal "in service" state, and (if its database and/or replica list are writable) start accepting updates.

stop

Stop the registry server that is running at *default_host*.

EXAMPLES

Start `rgy_admin`, list the replicas and their states, then set the default host to the master replica:

```
$ /etc/rgy_admin
Default object: rgy default host: dds://george
State: in service slave
rgy_admin: lrep -st
  (master) dds://martha state: in service 1987/11/16.12:46:59
           dds://george state: in service 1987/11/16.12:46:59
           dds://thomas state: in service 1987/11/16.12:46:59
rgy_admin: set -m
  Default object: rgy default host: dds://martha
  State: in service master replica list is writeable
```

NAME

`rgy_create` – registry creation utility

SYNOPSIS

`rgy_create`

DESCRIPTION

The `rgy_create` tool creates a new registry database on the local node, initialized with reserved names and accounts. It ordinarily should be run only once at a site. Replicas of the database are created by running `rgyd` with the `-create` option.

You must be root to invoke `rgy_create`.

Note that to convert a pre-SR10 registry to SR10 format, you should run only the `cvtrgy` tool, and you will never need to use `rgy_create`.

NAME

`rgy_merge` – merge registry database

SYNOPSIS

`rgy_merge -from //site [{ -merge | -compare } -verbose]`

DESCRIPTION

The `rgy_merge` utility merges the contents of two registry databases, a source database and a target database. You typically use it when you are joining two networks that have been operating separately and you want to combine their registry databases.

You must invoke `rgy_merge` while logged in as root at the master registry node for the target registry. Use the required `-from //site` argument to specify the master registry node for the source registry. The merge takes less time if the source database is smaller than the target database.

After you invoke `rgy_merge`, the tool prompts you to "login" with an account that owns the source registry.

Without a `-merge` or `-compare` option, `rgy_merge` attempts to add each entry in the source database to a copy of the target database and reports any conflicts in names or UNIX numbers. If there are no conflicts or errors, the tool asks whether you want to actually update the target database. If you respond affirmatively, it performs the merge on the target database and makes all replicas of the source registry slave replicas of the target registry.

If you specify `-merge`, `rgy_merge` performs the merge without querying you, provided there are no conflicts or errors. If you specify `-compare`, `rgy_merge` only checks for conflicts and does not perform a merge even if there are none.

The `-verbose` option causes `rgy_merge` to generate a verbose transcript of its activity.

NAME

rgyd – network registry server

SYNOPSIS

/etc/rgyd [**--create** | **--recreate** | **--restore_master**]

DESCRIPTION

rgyd is the network registry daemon. It manages all access to the network registry database. You must be the super-user to invoke **rgyd**.

The daemon can be replicated, so that several copies of the database exist on a network or an internet, each managed by a **rgyd** process. Only one registry daemon, the master, can accept operations that change the database (such as adding an account). If the daemon is replicated, the other replicas are slaves, which accept only lookup operations (such as validating a login attempt).

A Local Location Broker daemon (**llbd**) must be running on the local node when **rgyd** is started. Typically, both daemons are started at boot time from **/etc/rc**. The server will place itself in the background when it is ready to service requests.

OPTIONS

- create** Create a replica of the network registry. This option creates a copy of the registry database and starts a slave server process. You use **--create** only the first time you start a slave server process on a node. When you restart the daemon, you do not need any options at all. To create the master replica, use either **cvtrgy** (if you are converting an SR9 registry to SR10 format) or **rgy_create** (if you are creating a new SR10 registry).
- recreate** Recreate a slave replica. You should use this option only if a slave's copy of the database has been irreparably corrupted. It destroys the existing database and creates a new one.
- restore_master** Restart a master server and reinitialize all slave replicas. You should use this option only to recover from a catastrophic failure of the master node, (for example, if the database has been corrupted and then restored from a backup tape).

EXAMPLES

All of the commands shown in these examples must be run by root.

1. Start the master replica of the registry after you have created the master database via **rgy_create** or **cvtrgy**:

```
$ /etc/server -p /etc/rgyd
```

2. Start a slave replica of the registry.

```
$ /etc/server -p /etc/rgyd --create
```

3. Restart an existing replica (master or slave) of the registry.

```
$ /etc/server -p /etc/rgyd
```

4. Restart an existing replica of the registry on the remote host //yak.

```
$ /etc/crp -on //yak -cps -n rgyd //yak/etc/rgyd
```

NAME

rldev – release device acquired with **aqdev**

SYNOPSIS

rldev {*unit_number* | **-all**}

DESCRIPTION

rldev releases one or more devices acquired by **aqdev** (**acquire_device**). This command is valid only if our General Purpose Input/Output (GPIO) software package is running on your node.

NOTE: **aqdev** invokes a new shell. To release a device acquired in this manner, type CTRL/Z, which causes the shell to stop and **aqdev** to release the device.

ARGUMENTS

unit_number (optional)

Specify the unit number of the device to be released. Default if omitted: use **-all** below

OPTIONS

-all Release all devices acquired by the current process.

EXAMPLES

Release device 0.

```
$ rldev 0
Device 0 released.
$
```


NAME

rtchk – test traffic between adjacent routers

SYNOPSIS

/etc/rtchk [options]

DESCRIPTION

rtchk performs a simple test to verify that the router is able to pass packets to and from an adjacent router. **rtchk** is for use in a Domain internet.

Use the **-device** option to specify a network controller to test. You must give a device type (for example, RING, IIC) to the device option. The **rtsvc** program, with no command-line options, shows you which network devices your node has.

Older versions of **rtchk** used a different command-line syntax to specify the type of network hardware checked. The old command-line options still work in **rtchk** version 10.1, but are no longer supported.

For more information on **rtchk**, see *Managing Domain Routing and Domain/OS in an Internet*.

OPTIONS

-n *net.node_id* Test packet transmission to and from the specified node. The network id *net* must be a network that the router touches. If you use **-n** without **-dev**, you must specify a *net.node_id*. If you use **-n** with **-dev**, you must specify only the *node_id*, without the network number.

-dev[ice] *dev-name* [*dev-num*] Test packet transmission over a specific network device. Use the **rtsvc** program to display the names (used for *dev-name*) and controller numbers (used for *dev-num*) of the network devices attached to your node.

If you do not also specify a **-n** option, **rtchk** broadcasts its test packets. If the network contains more than one other node, **rtchk** receives more responses to its test packets than expected and prints warning messages. If you specify a **-n** option with the **-dev** option, **rtchk** sends the test packets only to the node you specify.

-s *n* Specify the number of test packets to exchange with the other router. If **-s** is not specified, 10 packets are exchanged.

-dat (default) Specify that each test packet carries 1024 bytes of test data.

-nodat Omit test data from the test packets.

EXAMPLES

Exchange 1000 test packets with node 4851 on network 3CE02A8. The router must be attached to that network.

```
$ /etc/rtchk -n 3CE02A8.4851 -s 1000
```

Exchange 10 short test packets with the other node attached to the IIC or T1 connection.

```
$ /etc/rtchk -nodat
```

Exchange 100 test packets with the other node on the IIC or T1 network.

```
$ /etc/rtchk -dev iic -s 100
```

Exchange 10 test packets with node 666 on the ring network.

```
$ /etc/rtchk -dev ring -n 666
```

SEE ALSO

More information is available. Type

help rtsvc For information on listing networks which touch your node

NAME

rtstat – display internet router information

SYNOPSIS

/etc/rtrstat [options]

DESCRIPTION

rtstat shows the behavior of an internet router at each of its network ports. rtstat is used in Domain internets. However, it can provide information about non-routing nodes as well as routing nodes.

For more information on rtstat, see *Managing Domain Routing and Domain/OS in an Internet*.

OPTIONS

-dev Report device-specific statistics for each port.

-net [net_id ...]

Report counts of references to each network specified. The reference counts for each network are roughly proportional to the number of packets transmitted towards the network, but may be somewhat higher. -net with no arguments uses the list of visible networks.

-r [n] Repeat every n seconds. If n is omitted, repeat every 10 seconds.

-n node_spec ...

Report statistics for each node in the list.

See help node_spec for details about node specification syntax.

If this option is omitted, rtstat reports statistics for the local node only.

-desc[ribe] Print a description, several lines long, of each statistic. The description appears only once for each statistic, the first time it is printed with a nonzero value.

EXAMPLES

1. \$ /etc/rtrstat

```
-----
1232.3D9      pkts routed:  110024  queue oflo:    0
               misrouted:      0      rt too far:   14

      RING      pkts sent:      73278  pkts rcvd:    72434

      IIC      pkts sent:      67830  pkts rcvd:    61077
```

2. `$/etc/rtstat -net`

```
-----
1232.3D9      pkts routed:  110024  queue oflo:    0
              misrouted:    0      rt too far:   14

      RING      pkts sent:    73278   pkts rcvd:    72434
              towards net:  1232   ref cnt:     74540

      IIC      pkts sent:    67830   pkts rcvd:    61077
              towards net:  1234   ref cnt:    53532
              towards net:  1231   ref cnt:    9193
              towards net:  1233   ref cnt:    5105
```

SEE ALSO

More information is available. Type

`help netstat`

For information about displaying other kinds of node behavior

NAME

rtsvc – set or display internet routing service

SYNOPSIS

`/etc/rtsvc [-device dev-name [dev-number] [options]]`

DESCRIPTION

rtsvc displays or alters the characteristics of a network port. **rtsvc** is used in Domain internets. You must be logged on to the node you wish to control in order to use **rtsvc**.

For complete information on **rtsvc**, see *Managing Domain Routing and Domain/OS in an Internet*.

OPTIONS

If no options are specified, **rtsvc** displays the characteristics of every active network. If you specify any other options, you must specify the type of network controller by using a `-device` command-line option.

You may use only one `-device` option on any command line:

`-dev[ice dev-name [dev-number]`

Specify the network device type: RING, IIC, or USER (for EtherBridge routers). The device number applies only to USER devices. You may use the name ETHERBRIDGE in place of USER if you prefer. The *dev-number* option applies only to USER networks, and is required. Find the device number by using **rtsvc** without command line options (as shown in the examples).

Earlier versions of the **rtsvc** command used a different command-line syntax for specifying network devices. The old command lines still work, but you should start using the new `-device` command lines as soon as possible. Future versions of **rtsvc** will not accept the older command lines.

This option changes the network ID of any network port:

`-net net_id` Assign the port a hexadecimal network ID number.

Note: If you use this option to change the network ID of a port on an active router, other nodes on the network can stop communicating with each other. Use this option only as directed in *Managing Domain Routing and Domain/OS in an Internet*.

You can specify only one of the following options at a time:

- route** Allow routing service to or from the port.
- noroute** Allow normal Domain/OS requests but no routing service.
- off** Do not allow Domain/OS requests or routing service.
- user *nn*** Set an EtherBridge network. The value is not changed until the routing node is rebooted or the routing process is stopped and restarted.

EXAMPLES

```
$ /etc/rtsvc -device iic -net 007302ED -route
```

Assign a network ID to the Interphase controller and allow internet routing at that port.

```
$ /etc/rtsvc -dev ring -noroute
```

Stop internet routing through the ring port, but allow normal Domain/OS is requests for paging, file service, etc. Do not change the node's network ID:

```
$ /etc/rtsvc
```

Display the networks attached to this node.

Controller	Net ID	Service offered
RING	76A0	Own traffic only
USER 46	768C	Port not open

The node in the last example touches two networks: a Domain ring and an ETHER-NET, via the EtherBridge product. You need the device number information ("46") from this display in order to turn on routing at the EtherBridge network. Use the device number as shown here:

```
$ /etc/rtsvc -dev user 46 -route
```

"46" is the device number.

SEE ALSO

- More information is available. Type **help netsvc** For information about controlling a node's network access

NAME

rwmt – read/write foreign magtapes

SYNOPSIS

rwmt [*option*]... [-p] {-r|-w|-i|-l} [*pathname*]...

DESCRIPTION

rwmt reads tapes from non-Domain installations and writes tapes that can be read by non-Domain installations. **rwmt** can read and write unlabeled tapes, as well as ANSI level 1–4 labeled tapes.

pathname (optional) Specify the name of file to be read from or written to tape. This argument is valid only with the **-r** and **-w** mode-control options (below). Multiple pathnames are permitted. Wildcarding is permitted for write (**-w**) operations only.

Default if omitted: read pathnames from standard input

OPTIONS

Mode control

You must specify one of the following mode-control options. If you omit this option, **rwmt** prompts you for it. The **-p** option tells **rwmt** to prompt for all necessary options.

-l[*label*] Write ANSI X3.27–1978 volume label on a tape. This option causes **rwmt** to write an ANSI volume label and dummy file on the magtape volume. You may specify an optional owner and volume ID, which are stored in the volume label. (see **-vid** and **-own** below. This is the way to initialize a labeled tape; if any information existed on the tape, it is erased by this labeling operation.

If you are labeling a tape, you can also use the following two options.

-vid *vol_id* Specify a 1–6 character volume ID for use when labeling a volume. This option is valid only when used with the **-l** mode-control option (above). The default volume ID is ' ' (blank).

-own *owner_id* Specify a 1–14 character owner ID for use when labeling a volume. This option is valid only when used with the **-l** mode-control option (above). The default owner ID is ' ' (blank).

-i[*index*] List objects on an ANSI-labeled physical tape volume. **-index** produces a listing of all files or file sections on an ANSI-labeled

physical tape volume. The contents of the physical volume (VOL1) label and all file header labels are written to standard output.

-w[rite] Specify one or more disk files (*pathname* argument) to be written to tape. The default format is ANSI labeled, ASCII, fixed-length records of 80 bytes each, and 80-byte blocks. If desired, any of these parameters can be changed using the options described below. If more than one *pathname* is specified, the disk files are written to sequential tape files. Tapes written by *rwmt* are always in accordance with ANSI level 4 format. Before writing a labeled file, the tape volume itself must be labeled with the **-label mode-control** option (above).

-r[ead] Specify one or more tape files to be read from tape and stored on disk. *read* reads one or more tape files and writes them to disk using the specified *pathnames* (*pathname* argument). The default tape file format is the same as that for the write option. If the tape is labeled under ANSI level 2, 3, or 4, the file format (block length, record length, and record format) is read from the tape. If the tape is unlabeled, or labeled with ANSI level 1, you must specify the tape format using the options below. If more than one *pathname* is specified, adjacent tape files are read and stored under the specified *pathnames*.

Label Control

-ansi (default) Specify that the tape is labeled in conformance to ANSI X3.27-1978, level 1, 2, 3, or 4.

-unlab Specify that the tape is unlabeled. Data spanning physical volumes is not supported on unlabeled tapes.

-asc (default) Specify that all tape file contents are in ASCII characters.

-ebc Specify that all tape file contents (except labels) are in EBCDIC characters.

-raw Specify that all tape file data is to be treated in raw form.

-npar (default) Specify no disturbance of parity bits when reading or writing data.

-par Specify that parity bits should be forced off when reading data from tape and forced on when writing data to tape.

-rl *reclen* Specify the maximum length, in bytes, of a record in the tape file. This option is valid only when used with either the **-r** or the **-w** mode-control options (above). It is unnecessary when reading an ANSI level 2, 3, or 4 file. The default record length is 80 bytes.

- bl *blocklen*** Specify the length, in bytes, of a physical tape block. This option is valid only when used with either the **-r** or the **-w** mode-control options (above). It is unnecessary when reading an ANSI level 2, 3, or 4 file. The default block length is 80 bytes.
- bf *blockfac*** Specify a blocking factor — the number of records to store into or read from a physical tape block. This is an alternative to the **-bl** option, since the record length multiplied by the blocking factor yields the block length. This option is valid only when used with either the **-r** or **-w** mode-control options (above). Using this option is meaningful only if your tape has fixed-length records. This option is unnecessary when reading an ANSI level 2, 3, or 4 file. The default blocking factor is 1.
- rf *format*** Specify record format. Valid values for *format* are **f** (fixed-length records and blocks); **d** (variable-length records (this is ANSI 'D' format)); **s** (spanned records); or **u** (undefined record format). The default format is **f**. Note that if you are writing a cartridge tape, only 512 byte blocks may be written; **d**, **s**, and **u** formats are not supported.

Tape File Identifiers

- fid *file_id*** Specify a 1–17 character file ID to be written in the file header label for use when writing a file to a labeled volume. This option is valid only when used with the **-w** mode-control option (above). If this option is omitted, the name of the file being written is used.
- f [*position*]** Specify the file position for **-r** or **-w** operations. Valid values for *position* are **cur**, **end**, or a nonzero integer value. A position of **cur** selects the current tape position; the tape must have been previously read or written by **rwmt** and its position must not have been disturbed. This option is valid only when used with either the **-r** or the **-w** mode-control options (above).

A position of **end** selects the end of the tape file set. This option is valid only when used with the **-w** mode-control option, and causes **rwmt** to append the specified disk file (*pathname* argument) to the very end of the tape file set.

A position specified by a nonzero integer value selects the file at that absolute position in the tape volume. This option is valid only when used with either the **-r** or **-w** mode-control options (above). If multiple *pathname* arguments are supplied, the value of *position* is incremented by one after each file has been read or written.

The default value for *position* is 1.

Backup Device Control

-dev d[*unit*]

Specify device type and unit number. *d* must be either *m* (for reel-to-reel magnetic tape), *ct* (for cartridge tape), or *f* (for floppy), depending on which drive is being used. *unit* is an integer (0-3). Both are required for reel-to-reel tapes (that is, -dev *m2*). A unit number is not required for floppy disks and cartridge tapes (that is, -dev *f*). If this option is omitted, *rbak* assumes device *m0*.

-nobs

Specifies that byte swapping should not be done in software. This operation is valid for magnetic tapes only. On the multibus data gets byte swapped. *rwmt* does byte swapping in software so that the tape gets written out in the correct machine order. *wbak* and *rbak* do not do byte swapping in software, as a result the two swaps done by the multibus cancel out. This option is useful in writing to a magnetic tape an intermediate file to which *wbak* output has been directed. Byte swapping should not be done by *rwmt* if the intermediate file written by *wbak* is now written raw to the magnetic tape using *rwmt*.

-reten

Retension the cartridge tape (unwind to the end, then rewind). This can be helpful if you have encountered cartridge-tape reading errors. Retensioning requires about 1.5 minutes to complete.

-nreten (default)

Do not retension the cartridge tape.

Miscellaneous Control Options

-sbin

Cause all stream files written to contain the binary attribute (normally, output stream files contain the ASCII attribute).

-p

Cause *rwmt* to prompt for all unspecified parameters.

EXAMPLES

Initialize a tape with the given owner and volume ID.

```
$ rwmt -label -own "R and D" -vid "demo"
```

Copy the wildcarded files to tape.

```
$ rwmt -w c?*_example -f 1 -rf d -rl 200 -bl 2048
32 records of "cmf_example" written to tape file 1.
8 records of "cmt_example" written to tape file 2.
4 records of "cpboot_example" written to tape file 3.
25 records of "cpf_example" written to tape file 4.
```

List the files on the tape.

\$ rwmt -index

Volume label:

Volume ID: "DEMO " Owner ID: "R AND D " Access: " "

File/Section		File ID	Cr Date	Acc	RF	RL	BL
1	1	CMF_EXAMPLE	83/02/17		D	200	2048
2	1	CMT_EXAMPLE	83/02/17		D	200	2048
3	1	CPBOOT_EXAMPLE	83/02/17		D	200	2048
4	1	CPF_EXAMPLE	83/02/17		D	200	2048
5	1	CPT_EXAMPLE	83/02/17		D	200	2048

End of file set.

\$

Copy tape file 3 to a disk file named `cpboot_example.tape`.

\$ rwmt -r cpboot_example.tape -f 3

4 records read from tape file 3 into

"cpboot_example.tape".

\$

`rwmt` permits a tape file to be read in "raw" mode. In this mode, each block read from the tape is written into one record in a stream file, unmodified by the program. To read a file in "raw" mode, you should specify the maximum record size using the `-rl` argument. If you do not, the default value of 80 bytes is used, and any records longer than that are truncated. Also, undefined record format should be used. For example

\$ rwmt -r -f 1 -rf u -raw -rl 512 rawfile

reads tape file number 1 into `rawfile`, with a maximum record length of 512 bytes.

Files may be written in the same manner:

\$ rwmt -w -f 1 -rf u -raw -rl 512 rawfile

The file `//backup/tmp1` is written out to the magnetic tape in "raw" mode. The record length is specified to be 8k and no byte swapping is done in software. This is useful for writing out an intermediate file to which `wbak` has written its output. Note that all tapes written by `rwmt` must have a ANSI standard volume label for `rbak` to be able to read the tape

```
rwmt -w -f 1 -raw -rl 8192 -nobs //backup/tmp1 -ansi
```

If `rwmt` writes a file with `-nobs` option, you should use `-nobs` option to read it using `rwmt`.

SEE ALSO

More information is available. Type

<code>help rbak</code>	For information on restoring or indexing a magnetic media backup file
<code>help wbak</code>	For information on creating a magnetic media backup file
<code>help media</code>	For information on removable media

NAME

salacl – salvage an access control list

SYNOPSIS

/etc/salacl [*options*] [*volume*]

DESCRIPTION

salacl salvages the (Access Control List (ACL) structure on the volume you specify. It merges duplicate ACLs into a single copy, and deletes unused ACLs. You should run salacl once a week or so unless you never receive reports that reference counts need repairing (that is, everything is perfect).

salacl cannot merge duplicate ACLs on files that are currently in use (locked) (for example, library files). This is not especially serious, as the duplication consumes very little disk space. To merge all possible ACLs on a node, including things like libraries, bring the node up diskless, mount the volume using mtvol, and then run salacl on the mounted volume.

volume (optional) Specify the entry directory pathname for the volume whose acls you intend to salvage. Note that salacl cannot salvage a volume mounted on a remote node.

Default if omitted: "/" node entry directory

OPTIONS

-n[o_s]ummary Suppress summary information.

-v[erify] Verify only; do not delete or merge any ACLs.

-n[o_m]erge Do not merge duplicate ACLs into a single copy.

-l[ist] List the UIDs of the ACLs that are being combined.

-u This option causes salvol to scan for files and directories with ACLs. When it encounters one, it puts out a line with UID of the ACL, followed by the UID of the file or directory. It also indicates the type of ACL, whether file or directory, and whether initial or not.

EXAMPLES

```
$ /etc/salacl
Warning: unable to merge two equivalent ACLs:
//grover/sys/node_data/boot_shell -
object is in use (OS/file server)
acl objects found:                24
acls merged with equivalents:     12
```

Salvage the ACLs of the current volume.

```
$ /etc/salacl -v
```

```
acl objects found:          24
acls which could be merged: 12
```

```
$ /etc/salacl -l
```

```
3A39A9F3.4000CBD6 39A77EF0.4000CBD6 (equiv acl)
3A75D6C2.E000CBD6 39A77EF0.4000CBD6 (equiv acl)
3AEE67CA.7000CBD6 39A77EF0.4000CBD6 (equiv acl)
3AEF69EE.E000CBD6 39A77EF0.4000CBD6 (equiv acl)
3AEFA618.7000CBD6 382E6E87.A000CBD6 (equiv acl)
ACL objects found:          43
ACLs merged with equivalent: 5
```

```
$ /etc/salacl -v -u
```

```
3AE41FE2.D000CBD6 393D4261.B000CBD6 (dir)
3AE41F4B.5000CBD6 380A1479.1000CBD6 (dir)
3A39A9F3.4000CBD6 3A39A9F2.3000CBD6 (file)
3AE41F85.0000CBD6 380A1472.F000CBD6 (dir)
3AE41F84.F000CBD6 380A1472.F000CBD6 (dir def)
3AE41F81.E000CBD6 380A1472.F000CBD6 (file def)
```

NAME

sald – salvage a directory

SYNOPSIS

sald *pathname*...

DESCRIPTION

sald corrects problems in directories caused by a system crash or network failure. The specified pathname must refer to a directory.

sald makes the specified directory usable and saves as much information as possible.

The following are symptoms of damaged directories:

- **ld** reports that the directory is empty, but it cannot be deleted.
- **ld** lists a file in the directory, but no other command can find the file.
- The directory is completely unreadable, and errors occur on every access attempt.

ARGUMENTS

pathname (required) Specify name of directory to be salvaged. Multiple pathnames and wildcarding are permitted.

OPTIONS

sald has no unique options.

EXAMPLES

Salvage the directory specified.

```
$ sald /sqh/data_dir
```

NAME

salvol – verify and correct allocation of disk blocks

SYNOPSIS

/etc/salvol [options] [lv_num] (from shell)
 ex salvol (from mnemonic debugger)

DESCRIPTION

Each logical volume is divided into disk blocks. salvol verifies and, if necessary, corrects the tables that describe the allocation of disk blocks to the files stored on the disk. salvol also returns to the free space pool all blocks that are no longer in use: those allocated to temporary files or to deleted portions of permanent files.

salvol can usually restore a disk after a system crash or an improper unmounting of a volume.

If no options are specified on the command line, then salvol prompts for all required arguments and options.

lv_num (optional)

A decimal value for which logical volume number to salvage.

OPTIONS

- a Read all blocks in all files. This option will take longer to run and is useful for finding block header errors or file blocks that cannot be read. This option is not useful for finding multiply allocated blocks or general disk problems.
- c *c*[*cnum*:*d_num*]
 controller type:
c = {w, s, f} (for winchester, storage module or floppy)
cnum = controller number, if specified, must be followed by a ':'
d_num = drive number

 This flag must be specified if any command-line options are used.
- f Fix error without prompting. (The default is to prompt.)
- h Print help text.
- n Check disk; only salvage if disk needs salvaging. This option is useful in scripts that mount secondary disks at boot time.
- p Polite mode; pause before overflowing screen. (The default is offline.)
- s Show some file statistics at completion.
- t Terminal mode, do not pause during output. (The default is online.)
- v Verify only; do not write anything to disk.

NAME

scrattr – screen attributes

SYNOPSIS

scrattr [**-avcpxy**]

DESCRIPTION

Without any options, **scrattr** lists the X and Y dimensions of the display in pixels. Screen attributes are always listed in the same order: X coordinates, Y coordinates, number of planes, number of primary colors. This order is independent of the order in which the options are specified.

OPTIONS

-v(verbose)

Gives a description of each field, and the attributes on separate lines. Without the **-v** option, attributes appear on the same line separated by tabs. If any options other than **-v** are specified, only that combination of attributes will be displayed, and always in the order given above.

-a Displays all attributes.

-x Displays the X dimension of the display in pixels.

-y Displays the Y dimension of the display in pixels.

-p Displays the number of bit planes on the display.

-c Displays the number of primary colors on the display.

NAME

scrto – set/show screen timeout

SYNOPSIS

scrto [**-none**] [*n*]

DESCRIPTION

scrto sets or displays the number of minutes the system waits before it shuts off the display screen. It begins counting minutes after the last input event or window configuration change.

By default, the system waits 15 minutes before it shuts off the display. Domain/OS turns the display back on whenever it receives an input event from the keyboard or mouse or whenever the DM creates, pops, moves, or resizes a window.

n (optional) Set the number of Domain/OS minutes for to wait before it shuts off the display.

Default if omitted: display current timeout setting

OPTIONS

-none Disable automatic timeout; never turn off the display.

EXAMPLES

Show initial setting.

```
$ scrto
```

```
The screen timeout is set to 15 minutes
```

```
$
```

Set delay to 10 min.

```
$ scrto 10
```

```
$
```

NAME

select – execute a select statement

SYNOPSIS

```

select arg [mode]
  case arg [to arg]
    [command...]
  [case...
    command...]
  [otherwise
    command...]
endselect

```

DESCRIPTION

select allows you to build a control structure that executes commands according to the results of one or more Boolean tests. The shell uses each case clause to perform a separate Boolean test on the initial select argument. If the case argument is equal to the select argument, the result of the test is true and the command(s) within the case clause execute.

You may test multiple cases simultaneously. If you place several case clauses on the same line (or specify line continuation with RETURN, the cases are logically or'd and return true if any one of the cases is true. (See example below.)

The (optional) *to* clause allows you to specify an integer or string range for testing. For example, you might specify **case 0 to 9** to test for any single digit, or **case a to z** to test for a lowercase letter.

The (optional) **otherwise** clause executes if and only if none of the case clauses returns true (regardless of whether the selection mode was **oneof** or **allof**).

ARGUMENTS

arg (required) Any valid token, defined integer or string variable, or expression. **select** compares the first *arg* with each of the case *arg*'s to determine which command(s) to execute.

mode (optional) Specify selection mode. Valid modes are **oneof** and **allof**. If you specify **oneof** (the default), **select** executes only the first case statement that returns a true value. If you specify **allof**, **select** executes all case statements that return true.

Default if omitted: use **oneof**

command... (optional)

Specify the command to be executed when the case test returns true. This may be a shell command, a shell script, a variable assignment, or any other valid shell operation. Multiple commands are permitted; separate them with semicolons or newline characters.

Default if omitted: no command executed for this case clause

EXAMPLES

```
select ^a all of
  case 1 case 2 case 3
    args "This will print if ^a = 1 or ^a = 2 or ^a = 3"
  case 1 @
  case 2 @
  case 3
    args "This is the same test as the previous one, since the"
    args "carriage returns are escaped."
  case 4 # This is a case without a body to execute.
  case 5 to 10
    args "This will print if ^a is in the range 5-10."
  case 6
    args "This will also print if ^a = 6, since all of is"
    args "specified."
  otherwise
    args "This will print if ^a is not between 1 and 10."
endselect
```

NAME

`send_alarm` – send messages to alarm servers

SYNOPSIS

`send_alarm string ...` *[[target_option ...]]* `[-l]`

DESCRIPTION

`send_alarm` transmits one-line messages that the alarm server can display. You can direct messages to the following targets:

- A particular user, specified by the name used for login
- The user on a particular node, specified by the node's entry directory or node ID
- The users on all the diskless nodes that have a particular paging partner
- Combinations of these categories of users

You must always specify at least one user to receive the message, but you may use any of these techniques for choosing users.

Messages are not stored and message delivery is not guaranteed. If the intended recipient is not running an alarm server when the message is sent, or if other problems arise, the message is never delivered. `send_alarm` notifies you when messages cannot be delivered.

You can send up to 80 characters using `send_alarm`. If your message is longer, you should send it with the mail program.

ARGUMENTS

`string ...` (required) Specify the message you want to send. Multiple strings are permitted, separated by blanks. If several strings are present, they are concatenated with intervening spaces to form the message.

OPTIONS

`-l` List the target nodes or users as the messages are sent. You must specify at least one of the following target options.

`-u[ser] login_name ...` Specify one or more users to whom the message should be sent. If you use this option more than once, several lists of users are concatenated. Every user on the concatenated list receives the message.

If a user logged on several nodes at the same time, and has alarm servers running at each of those log-in sessions, only one of the alarm servers can receive the messages.

The `-user` option is not likely to work unless the user's home directory is an absolute pathname. For example, users with "/" as their home directory may not be reachable. A home directory like `//smith/jones` is more likely to work. Expect erratic results if multiple users share a single home directory.

`-n[ode] node_spec ...`

Transmit the message to whatever user is logged in on the specified node, if any. Type `help node_spec` for details about node specification syntax.

Multiple node specs are permitted; separate them with blanks. If you use this option more than once, several lists of nodes are concatenated. Every user on a node on the concatenated list receives the message.

If several users are logged in on one node simultaneously and are running alarm servers, only one user can receive messages directed to that node.

`-di[skless] node_spec ...`

Transmit the message to whatever user is logged in on any diskless node whose host node is specified by `node_spec` as in the `-node` option. Multiple `node_specs` are permitted. If you use this option more than once, several lists of nodes are concatenated. Note that this option does not send a message to the user on the paging partner. See `-din` below.

`-din node_spec ...`

This option allows you to send a message to a disked node and all of its diskless partners. This is the same as specifying `-node node_spec ... -diskless node_spec ...`.

`-me` or `-myu[ser]`

Adds you to the list of users to receive the messages.

`-myn[ode]`

Adds the node at which the command is entered to the list of target nodes.

`-mydi[skless]`

Same as `-diskless node_spec`, where `node_spec` is the node at which the `send_alarm` command was entered.

`-mydin`

Same as `-mynode -mydiskless`.

`-alln[ode]`

Send the message to every node seen by an `lnode`.

EXAMPLES

```
$ send_alarm 'Meeting is cancelled' -user joe_k sue_b john_f mary_g
```

```
$ send_alarm Please log out. Node AAD going down. -din 0aad
```

```
$ send_alarm Compilation completed -me
```

```
$ send_alarm Power going off at 17:30 tonight -allnode -l
```

```
$ send_alarm 'Loop 13 is being switched out' *loop13
```

Note: In the above example, `loop13` is a file in the working directory containing further input for the `send_alarm` command line. It might, for example, contain the text:

```
-n //spruce //eddie  
-n //top 317F  
-u network_manager  
-mydin
```

SEE ALSO

More information is available. Type

alarm_server For details about setting up the alarm server to receive messages and to send longer messages, with ensured delivery

NAME

server – run a server process

SYNOPSIS

/etc/server [-p] "command-pathname arg1 arg2 ..."

DESCRIPTION

The **server** command runs a program with an identity of user "user" and group "server" just as if the command had been started using the Display Manager's **cps** command. It also marks the new process in such a way that the server program will not be terminated by the Display Manager when the user logs out.

In addition to allowing users to start server processes, this command is useful for killing servers that are running with the identity **user.server**. For example,

```
$ /etc/server /bin/kill -9 1532
```

OPTIONS

-p The **-p** option preserves the current SID of the person issuing the command. Otherwise, */etc/server* sets the SID to **user.server.none** for the command.

NAME

set – set current shell conditions

SYNOPSIS

set [*options*]

OPTIONS

- b[on]** Send the output of a background process (created with the & parsing operator) to the display. The output of the background process is displayed in the transcript pad of the shell where it was invoked. If you do not specify **-b**, the output of the background process is sent to `/dev/null`.
- boff** (default) Do not display output from a background process.
- nb[on]** (default) Same as **-boff**.
- c[ommand]** *arg1* ... Execute the following argument(s) as a shell command, exactly as if it had been read as an input line. If any argument contains explicit blanks, enclose it in quotation marks. The shell passes all text following **-c** to *arg1* as arguments, so if you want to specify other options to the `sh` command itself, they must precede **-c**.
- e[on]** Enable evaluation of variables outside of expressions. If **-e** is specified, the shell always evaluates variables, regardless of the context in which they appear. If **-e** is not specified, variables are evaluated only inside variable expression delimiters, (*expression*); otherwise, the shell treats the `^var_name` expressions as strings and they are not evaluated.
- eoff** (default) Evaluate variables only inside expressions.
- ne[on]** (default) Same as **-eoff**.
- i[nter]** Behave as though input is being entered interactively: prompt for each input line, and do not exit on errors or quit faults (DQ or CTRL/Q from keyboard). Normally, the shell executes interactively only if its input comes from a pad or SIO-line. Use of this option forces prompting.
- s[cript]** (default) Behave as though executing a shell script: do not prompt and abort on error. A shell does not normally quit; any error or quit command is assumed to apply only to the last command given to the shell.
- ni[nter]** (default) Same as **-s**.
- n[execute]** Interpret each command line only; suppress execution.
- ex[ecute]** (default) Interpret each command line and execute it.

- p[rompt]l *prompt_string*** Define the prompt string for the shell created with `sh`.
- =-p[rompt] *subprompt_string*** Define the subprompt string for the shell created with `sh`. (The subprompt appears when you continue a shell command over more than one line).
- start [file] (default)** Execute the specified script after the shell is created. If *pathname* is not specified, the shell searches the log-in home directory for a file called `user_data/sh/startup` and executes it if it exists. No error occurs if that file does not exist.
- nstart** Disable start-up file execution.
- v[on]** Display each line of text in the transcript pad as it is read by the shell program.
- voff (default)** Disable input verification.
- nv[on] (default)** Same as `-voff`.
- x[on]** Display each command in the transcript pad immediately before execution. Each command is given in full, with its complete pathname and with the values of arguments inserted.
- xoff (default)** Disable input examination.
- nx[on] (default)** Same as `-xoff`.

EXAMPLES

```
$ set -p l 'Input> '
```

Change the current shell's primary prompt to `Input>` . Note the use of quotation marks to preserve the trailing blank.

```
$ set -eon -xon -von
```

Enable variable evaluation, command examination, and verification.

SET

Aegis

SET

SEE ALSO

More information is available. Type

help sh

For details about the shell

NAME

setvar – set the value of a variable

SYNOPSIS

setvar [*options*] {[-type *type*] *var_name value ...* | *variable_list*}

DESCRIPTION

The setvar command takes pairs of arguments, which may be preceded by an optional type-specifier (*-type type*).

By default, the type of each variable specified in the setvar command depends on the type of each input value. However, you can use the *-type* argument to specify the individual type(s) of the the variables.

NOTE

If a value is not of the type specified by the *-type* argument, an error results.

For more information on variables, refer to the manual *Using Your Aegis Environment*.

ARGUMENTS

variable_list (optional)

Specify the names of the variables that receive the input values.

Default if omitted: must specify *-type* (below)

OPTIONS

-type type var_name...

Specify the type of the input value(s) that can be assigned to the particular variable name(s). Multiple variable names are permitted, separated by blanks. Once you specify a type in a particular setvar command, setvar assigns that type to all subsequent variable names, until you change the type specification. Valid types are:

str[ing]	Character strings
int[eger]	Integer numbers
bool[ean]	Boolean values
env[ironment]	Environment variables
any	Any type (the default)

If the type of the input value does not match the type specified for that variable name, setvar issues an error and asks you to enter another input value. Use *-type any* to restore the shell to its default state. In this case, it determines the proper variable type automatically.

Specifying `-type env var_name` causes the variable to become an environment variable.

EXAMPLES

In the first example, we create several variables using `setvar` and then list them using the `lvar` command.

```
$ setvar i 1                # an integer
$setvar b true             # a boolean
$ setvar s1 string         # a string
$setvar -type string s2 true # a boolean forced into a string
$lvar i b s1 s2
integer i = 1
boolean b = true
string s1 = string
string s2 = true
```

In this example, we set several variables of different types and then list them.

```
$ setvar -type int i1 1 i2 2 -type str s1 3 s2 4 -type any i3 1
$ lvar i1 i2 s1 s2 i3
integer i1 = 1
integer i2 = 2
string s1 = 3
string s2 = 4
integer i3 = 1
```

The following is an error message example. In this case, we are trying to set an integer to a string.

```
$ setvar -type int z ten
?(sh) semantic error - 'ten' is not an integer.
```

SEE ALSO

More information is available. Type

- | | |
|----------------------|--|
| help eoff | For details about restricting variable evaluation to within variable expressions |
| help eon | For details about enabling global variable evaluation |
| help existvar | For details about existing variables |
| help export | For information about environment variables |
| help dlvar | For information about deleting a variable |
| help read | For information on assigning multicharacter strings to variables |
| help readc | For information on assigning whole-line strings to variables |
| help readln | For information on assigning whole-line strings to variables |

NAME

sh – invoke a shell, command line interpreter

SYNOPSIS

sh [*options*] [*pathname* [*arg* ...]]

DESCRIPTION

The shell has four types of commands:

External Commands

These are programs that reside on your disk. They are invoked when you type in their pathname or, if their directories are included in your command search rules, when you type their leafname.

Internal Commands

These are built-in shell commands (see below). The shell always looks for these first.

Control Structures

These are programming constructs that allow you to control the flow of control in a shell script. Note: Since these structures are legal anywhere on the command line, you must enclose them in quotation marks when using the **help** command (for example, **help 'if'**).

Expressions

These are delimited by '()' and '()'. Inside of these double parentheses you can set variables, compare values and perform other standard integer, string or Boolean operations. The assignment operation (*variable := value*) is a special case that does not have to be enclosed in double parentheses.

Any of these commands can have its output redirected or may be invoked in the background using the shell's parsing operators (>, >>, >?, >>? <, <<, <?, <<?, |, &...) See *Using Your Aegis Environment* for details.

Internal Commands

Flags von, voff, xon, xoff, bon, boff, eon, eoff

Variables

readc, read, readln, existvar, lvar, dlvar, setvar, export

Control Structures

if, while, select, for@* eqs, existf, return, exit, next, source, set, abtsev, not

Miscellaneous

args, csr, rdym, hlpver, inlib, umask

Expressions

true, false@* :=, or, and, =, <, >, <=, >=, <>, +, =, *, /, mod, **, (,), not

ARGUMENTS

pathname (optional) Specify a file containing a shell script to be executed. Each line in the file is interpreted as a shell command.

args (optional) Default if omitted: read standard input

Specify any arguments to be passed to the program in file *path-name*. Arguments are substituted for *^n* expressions in the program: *arg1* for *^1*, *arg2* for *^2*, etc. *^** can be used to specify all the arguments at once. (See the manual, *Using Your Aegis Environment* for details on passing arguments to shell commands.) See example 1 below.

Default if omitted: no arguments passed

OPTIONS

-b[on] Send the output of a background process (created with the *&* parsing operator) to the display. The output of the background process is displayed in the transcript pad of the shell where it was invoked. If you do not specify **-b**, the output of the background process is sent to */dev/null*.

-boff (default) Do not display output from a background process.

-nb[on] (default) Same as **-boff**.

-c[ommand] arg1 ...
Execute the following argument(s) as a shell command, exactly as if it had been read as an input line. If any argument contains explicit blanks, enclose it in quotation marks. The shell passes all text following **-c** to *arg1* as arguments, so if you want to specify other options to the *sh* command itself, they must precede **-c**.

If *arg1* is the name of a shell script, note that the script creates a new shell level for execution (just as if you had invoked it at the *\$* prompt). Thus activities in the script that are level-dependent (such as assigning values to shell variables) do not propagate upward when the script exits. This is in contrast to the **-start** option and the shell command source, which execute scripts at the current shell level.

-e[on] Enable evaluation of variables outside of expressions. If **-e** is specified, the shell always evaluates variables, regardless of the context in which they appear. If **-e** is not specified, variables are evaluated only inside variable expression delimiters, (*(expression)*); otherwise, the shell treats the *^var_name* expressions as strings and they are not evaluated.

-eoff (default) Evaluate variables only inside expressions.

-ne[on] (default) Same as **-eoff**.

- f[first])** Do not exit after executing the command given by the **-c** option. This option is valid only if **-c** has been specified, and must precede **-c** on the command line.
 - i[nter])** Behave as though input is being entered interactively: prompt for each input line, and do not exit on errors or quit faults (the Display Manager **dq** command or CTRL/Q from keyboard). Normally, the shell executes interactively only if its input comes from a pad or SIO line. Use of this option forces prompting.
 - s[cript] (default)** Behave as though executing a shell script: do not prompt and abort on error. A shell does not normally quit; any error or quit command is assumed to apply only to the last command given to the shell.
 - ni[nter] (default)** Same as **-s**.
 - n[execute]** Interpret each command line only; suppress execution.
 - ex[ecute] (default)** Interpret each command line and execute it.
 - p[rompt]1 *prompt_string*** Define the prompt string for the shell created with **sh**.
 - p[rompt]2 *subprompt_string*** Define the subprompt string for the shell created with **sh**. (The subprompt appears when you continue a shell command over more than one line).
 - start [*file*]** Execute the specified script after the shell is created. If *file* is not specified, the shell searches the log-in home directory for a file called **user_data/sh/startup** and executes it if it exists. No error occurs if that file does not exist.
- Note that the script is executed at the current shell level, so that level-dependent activities (such as assigning values to shell variables) persist when the script exits. This is in contrast to the **-c** option, which executes scripts at the next lower shell level.
- This option is a default if **sh** is the first program invoked in a new process (i.e., **cp /com/sh**). It is not a default at any other time (i.e., when you type **sh** at the dollar sign or call it from a script).
- nstart** Disable start-up file execution.
 - v[on]** Display each line of text in the transcript pad as it is read by the shell program.

- voff** (default) Disable input verification.
- nv[on]** (default)
Same as **-voff**.
- x[on]** Display each command in the transcript pad immediately before execution. Each command is given in full, with its complete pathname and with the values of arguments inserted.
- xoff** (default) Disable input examination.
- nx[on]** (default)
Same as **-xoff**.

EXAMPLES

```
$ sh program-name arg1 arg2 ...
```

The shell executes the commands in the file **program-name**, and substitutes the arguments (*argn*) for character sequences `^n` in the program file.

```
$ sh -n my_script
```

Interpret each line in **my_script**, but do not execute anything.

SEE ALSO

More information is available. Type

- shell** For general information about the shell
- shell commands** For an index of shell commands
- shell i_o** For a description of the shell input/output redirection operators

NAME

`show_lc` – shell script to indicate obsoleted system calls in a binary file

SYNOPSIS

`/etc/show_lc binary_file ...`

DESCRIPTION

This script will show which calls in an object module have been obsoleted and replaced. This will not show which procedure made the call. Generate an xref listing to do that. Nor will it show any use of the old `name_$name_t` or `name_$pname_t` data types. You must ensure that the object being examined is in fact an object module.

The *binary_file* argument is required.

NAME

shutspm – shut down SPM on a node

SYNOPSIS

shutspm

DESCRIPTION

When the server process manager (SPM) runs in place of the DM, it waits on the eventcount file `node_data/spmshut_ec`. **shutspm** advances this eventcount, causing the SPM to perform an orderly shutdown of the node.

To shut down the SPM with **shutspm**, create a remote process (via the `crp` command) on the target node and type **shutspm**.

Normally, only system administrators may shut down the SPM using this command. This is because SPM creates the `node_data/spmshut_ec` file with the following ACL (provided the default file ACL for `node_data` gives all rights to `%.%.%.%`):

Subject ID	Access Rights
	<code>%.sys_admin.% pgndwrwx</code>
<code>%.%.%.%</code>	<code>---d-r-</code>

This ACL limits **shutspm** shutdown to accounts with the `sys_admin` project name, but permits any account to delete the `spmshut_ec` file whenever SPM is not using it. If, however, the default file ACL for `node_data` has been changed, SPM creates the eventcount file using that default ACL. Note that a subject identifier must have at least `r` and `w` rights to shut down SPM.

If the `spmshut_ec` file already exists when SPM starts up, SPM does not change its ACL.

To prevent SPM from responding to the **shutspm** command, add the following line to the `node_data/startup.spm` file:

```
no_shutspm
```

EXAMPLES

```
$ crp -on 1fb -login sys_admin
```

Create remote process on server node `1fb` and log in with the system administrator account.

\$ shutspm

Shut down the SPM on server node 1fb.

SEE ALSO

More information is available. Type

help spm

For details about the server process manager

NAME

sigp – signal a process

SYNOPSIS

sigp [*process_name* ...] [*options*]

DESCRIPTION

sigp causes a quit or stop fault in a process. This is particularly useful for stopping background processes such as those created by the **cpo** (*create_process_only*) and **cps** (*create_process_server*) Display Manager commands.

You may discover which processes are currently active by using the **pst** (*process_status*) command.

ARGUMENTS

process_name (optional)

Specify name of process to be signaled. Multiple process names and wildcarding are permitted.

Default if omitted: **-uid** required (below)

OPTIONS

- q[uit]** (default) Cause a quit fault in the process (like the Display Manager command **dq** (CTRL/Q)). Executing programs halt, but the process remains active.
- s[top]** Ask the entire process to stop cleanly (closing streams, etc.).
- b[last]** Stop the process in the nucleus (don't go to user mode). This brings everything to a halt without letting the system attempt to clean up.
- c[ode] fault** Signal the process with the hexadecimal status code *fault*.
- uid high low@*** **-uid high.low**
Stop the process with the given UID. *high* and *low* indicate the two halves of the UID.
- l** List processes signaled.

EXAMPLES

Generate a quit fault in process_7, which will halt the program currently running there, but leave process_7 itself active.

```
$ sigp process_7 -quit
$
```

Stop process_7 completely.

```
$ sigp process_7 -stop -l
"process_7" stopped.
$
```

NAME

siorf – receive a file from a remote host

SYNOPSIS

siorf [*options*] [*pathname* ...] [*]

DESCRIPTION

siorf accepts remote host transmissions from the appropriate SIO line, decodes them according to the proper protocol, and writes them to the file you specify.

Arguments and options may appear in any order and are processed and take effect as encountered. This means options must be specified before the file(s) for which they are intended.

You do not need to use the **tctl** command to set the **sync** and **insync** parameters of the SIO line when receiving a non-ASCII file. **siotf** and **siorf** recognize the types of the files being transferred and set these parameters correctly.

ARGUMENTS

pathname (optional) Specify the name of the file to receive the transmission. If you omit the filename, **siorf** waits for the host to specify a receive file. (If you want the transmission written to standard output, use the ***** option.) **siorf** terminates when it receives an end-of-transmission (EOT) signal, unless you include the **-f** option.

Default if omitted: see above

OPTIONS

- l** *n* Specify SIO line being used for the transmission. The default SIO line is 1.
- n** Select the Nibble protocol.
- f** Cause **siorf** to monitor the SIO line for host transmissions until it receives an error message over the SIO line or CTRL-Q from the node. When you include this option, EOT does not cause **siorf** to terminate.
- r** Replace file(s) if they already exist.
- ae** Abort on error. Otherwise, transmission continues until EOT.
- x** *host_file* Request a remote host file to be transmitted. This presumes a host counterpart of **siotf** (**sio transmit file**) is active.
- *** Receive transmission to standard output. This option is valid only if the *pathname* argument is omitted.

EXAMPLES

Create (or replace) a binary file `prog.bin` with the data received over SIO line 2; presumably being sent by an `siotf` counterpart.

```
$ siorf -l 2 -r prog.bin
```

Receive files over SIO line 1 whose names are specified by the transmission side (host or Domain node). Existing files are replaced if needed. `siorf` remains active until CTRL/Q or error occurs.

```
$ siorf -r -f
```

Request file `ask_file` to be sent over SIO line 1 and write data received to `/eng/new_copy`. Presumes the other side is running `siotf` or equivalent in "forever" mode.

```
$ siorf -x ask_file /eng/new_copy
```

SEE ALSO

More information is available. Type

`help siotf` For details about transmitting a file to a remote host

NAME

siotf – transmit a file to a remote host

SYNOPSIS

siotf [*options*] [*pathname ...*] [***]

DESCRIPTION

siotf sends the Domain file(s) you specify to a remote computer ("host") using the appropriate serial input/output (SIO) line and protocol.

Arguments and options may appear in any order and are processed and take effect as encountered. This means options must be specified before the file(s) for which they are intended.

You do not need to use the **tctl** command to set the **sync** and **insync** parameters of the SIO line when receiving a non-ASCII file. **siotf** and **siorf** recognize the types of the files being transferred and set these parameters correctly.

The transmission protocols used by **siotf** are described in the section on protocols, below.

ARGUMENTS

pathname (optional)

Specify the name of the file to be transmitted. If you wish to transmit data from standard input, use the ***** option.

Default if omitted: must use *****

OPTIONS

- l n** Specify the SIO line to be used for transmission. The default SIO line is 1.
- n** Select the Nibble protocol. (See the "Protocols" section, below.)
- f** Cause **siotf** to continue monitoring the SIO line for transmission requests from the remote host rather than terminating when transmission is complete.
- obj** Obsolete option. At SR9.5, **siotf** automatically detects binary objects and transmits them properly. Prior configuration of the SIO line (via the **tctl** command) is no longer necessary.
- ae** Abort on error rather than attempting to continue.
- x host_file** Pass a filename to the remote host. The host can use this name for the next file it receives from the node. This presumes a host counterpart to **siorf sio_receive_file** is active.
- *** Read from standard input and send standard input to the remote host. Signal end of data with an end of file (CTRL/Z).

EXAMPLES

Wait for file requests over SIO line 1 and transmit them.

```
$ siotf -f
```

Transmit file `prog1.bin`, then transmit file `prog2.bin` over SIO line 2.

```
$ siotf -l 2 prog1.bin prog2.bin
```

Send the name `tell_file`, then transmit the file `/eng/notes`. Presumably the receiving side is in "forever" mode (`-f`) and

```
$ siotf -x tell_file /eng/notes
```

is thus waiting for instructions.

PROTOCOLS

To permit binary and ASCII file transmissions, we have implemented two protocols: Plain and Nibbled.

Plain

Plain protocol is the default. It assumes that the host operating system can transmit and receive all 256 bit patterns, so there is no need to use escapes or to nibble at the ASCII or binary files. Even if the host can handle only the ASCII character set, you should use the Plain protocol for transmitting ASCII files.

The format of this protocol is as follows:

```
STX type COUNT...data...CHECKSUM CR
```

where:

STX is the standard ASCII STX (02).

type Is a small ASCII letter that identifies the record type as follows:

a	ACK
d	DATA
e	EOF
h	HELLO
g	ANSWER_HELLO
n	NAK
p	PARTIAL
t	TYPE
x	NAME

z EOT
? ERROR MESSAGE

COUNT is the number of data bytes in the record (not to exceed 255), nibbled and transmitted as two ASCII bytes (@ and the capital letters A through O).

CHECKSUM Is a 1-byte calculated checksum, nibbled and transmitted as two ASCII bytes.

CR Is a standard ASCII carriage return.

The "t" and "p" types are provided primarily for file transfers occurring between Domain nodes. "t" informs siorf of the type of file being transmitted. In this case, the *data* field consists of a single character that identifies the type of file as follows:

u uasc file (normal ASCII text file)
o Domain object file
m Non-streams file, accessed though the mapping primitives
r Streams record file

If you are transmitting a streams record file (type "r" above), the protocol now offers the "p" message type, which siorf uses to transmit partial records to siorf. siorf can transmit at most 255 bytes at a time. If a record is larger than 255 bytes, transmission occurs in 255-byte pieces; all but the last piece has the "p" type message. siorf transmits the last piece using the normal "d" type message so that siorf recognizes it as the end of the record.

Nibbled

If the host cannot send or receive anything but ASCII characters, use Nibbled protocol to transmit non-ASCII data. Transmitted records use a record format identical to that of Plain protocol, except that ^S replaces the STX byte. For siorf, each byte from the file is nibbled into two ASCII characters (@ or A through O). For siorf, the low four bits of each two bytes received are concatenated; this protocol checks the ASCII range of the received bytes. A byte out of range causes siorf to send the host an NAK signal. A byte out of range in five consecutive records causes siorf to issue an error message, and terminate. The count field of nibbled records contains the original count (that is, the number data bytes before nibbling). To select the Nibbled protocol, use the -n option with siorf or siotf.

When you execute siorf, it issues the hello record to signal that it is there, and to clear any transmission that may have preceded your command. It expects to receive the answer_hello response. siotf also does this before it begins transmitting records.

siorf acknowledges each remote host transmission. siotf waits for the host to acknowledge each transmission. These acknowledgements have the format:

STX a CR (or) STX n CR

STX is either STX or \bar{S} , and a and n are the small letters a (ACK) and n (NAK). The programs recover from a NAK by retransmitting the record in question. After ten consecutive unsuccessful retries, the programs issue an error message and abort. All messages must be acknowledged, including error messages.

The end of file signal is a record with the following format:

STX e CR

where e must be the small letter e. Host programs should acknowledge the EOF signal.

The end-of-transmission signal is a record with the following format:

STX z CR

where z must be the small letter z. Host programs should ACK the EOT signal. If the programs do not receive transmissions or ACKs for 60 seconds, they issue time-out error messages and terminate.

NOTES

siotf opens a stream to its SIO line in "cooked" mode, siorf opens the stream in "raw" mode. Both programs synchronize with host XON/XOFF (CTRL/Q, CTRL/S) signals.

If you specify a Domain file that cannot be opened, the programs issue an error message. If a file specified by a record received from the host cannot be opened (or created), the programs issue an error message, and transmit an error message to the host. However, they continue processing their parameters or (if you specified $-f$) waiting for host requests.

siotf does not transmit EOT if you specify $-f$. siorf does not terminate at EOT if you specify $-f$. If you omit $-f$, siorf waits until it receives an EOT signal from the host, or times out.

The programs accept type "?" error messages instead of ACK or NAK signals. The programs display the error messages, and terminate (even if you specified $-f$). If siorf gets an error message while receiving a file, it aborts. If you included $-f$, the programs try to remain active as long as possible.

Model programs to serve as the host-side counterparts to siorf and siotf have been supplied in /sys/source/emt. These are models in FORTRAN and in Pascal. The FORTRAN subroutines that need to be modified for host-specific use are in /fBhost_model_subsl.fin. The Pascal procedures to be modified are clearly marked in the Pascal model programs. For a particular host environment. You may also need to modify other areas of these models.

SIOTF

Aegis

SIOTF

SEE ALSO

More information is available. Type

help siorf

For details about receiving a file from a remote host

NAME

source – execute a shell script at the current shell level

SYNOPSIS

source *script_name* [*arg1...*]

DESCRIPTION

source allows you to execute a shell script at the current shell level. When you type

```
$ my_script arg1
```

your script runs in a new (subordinate) shell level. This means that all variables are now defined at a new level; that your script can't delete or otherwise affect variables at the level above; that state settings like **von/bon/eon** that the script sets vanish when the script finishes, and so forth.

On the other hand, typing

```
$ source my_script arg1
```

executes the script at the current shell level, just as though you had typed the contents of **my_script** into the process input window (and filled in the command-line arguments yourself). If the script says **von**, then **von** is set after the script exits. If it defines a variable, that variable still is defined, etc.

ARGUMENTS

script_name (required) Specify the name of the script to be executed.

arg1... (optional) Specify any arguments to be passed to the script.

Default if omitted: no arguments passed

NAME

srf – sort and/or merge text files

SYNOPSIS

srf [*options*] [*pathname ...*]

DESCRIPTION

srf sorts lines of all the named files together and writes the result on the standard output.

The sort key is an entire line. Default ordering is alphabetic by characters as they are represented in ASCII format (digits, then uppercase characters, then lowercase characters, then special characters).

ARGUMENTS

pathname (optional)

Specify file(s) to be sorted. Multiple pathnames are permitted.

Default if omitted: read standard input

OPTIONS**Sort Key Control**

Use one of the following:

- b Omit leading blanks from keys.
- s *n* Sort based on the subfield starting in column *n*. If this option is omitted, sorting starts in column one.
- f Fold all letters to a single case.

Input Character Control

Use one of the following

- d Use dictionary order: only letters, digits, and blanks are significant in comparisons. Special characters (punctuation, control characters, etc.) are ignored.
- i Ignore all nonprinting, nonblank characters.

Sort Mode Control

Use one of the following:

- m Merge only; the input files are already sorted.
- r Reverse the sense of the sort; list output entries in reverse order.

EXAMPLES

List contents of current working directory in order of date last modified.

```
$ ld -c -dtm | srf
```

List most recently changed files first.

```
$ ld -c -dtm | srf -r
```

List files by size with largest files first.

```
$ ld -c -bl | srf -r
```

NAME

stcode – translate status code value to text message

SYNOPSIS

stcode *hex_stat_code*

DESCRIPTION

stcode prints the text message associated with a hexadecimal status code. This command is useful when a user program produces a hexadecimal status code instead of the textual message.

stcode processes predefined status codes. No provision is currently made to add user-defined status codes to the error text database.

hex_stat_code (required) Specify hexadecimal status code to be translated.

EXAMPLES

```
$ stcode 80001
```

```
disk not ready (from OS / disk manager)
```

NAME

subs – set or display subsystem attributes

SYNOPSIS

subs *object* [*subs_name*] [*options*]

DESCRIPTION

subs is used to set or show protected subsystem attributes. When setting subsystem attributes, you must be running in that subsystem.

ARGUMENTS

object (required) Specify pathname of an object. The function of the object (either a protected file or a managing program) is determined by options described below.

subs_name (optional) Specify name of a subsystem. The shell searches the directory `/sys/subsys` for the specified subsystem. If you specify this argument, the attributes of the named subsystem are set as directed by the options described below.

Default if omitted: display attributes of *object*

OPTIONS

- data** Set or display the name of the subsystem that manages *object*.
- mgr** Set or display the name of the subsystem for which *object* is a manager. Object must be an executable file (a program).
- up** Increase the privilege level of a process running in a subsystem so that it can directly access the objects it owns.
- down** Decrease the privilege level of a process; opposite of **-up**.
- l** List subsystem attributes and/or manager fields. This is the default action if *subs_name* is not specified.
- br** Display only the name of the subsystem. Not valid if attributes are being set.

EXAMPLES

The following example illustrates the use of protected subsystems. First we show a Pascal source program written to manage the subsystem. (The calls issued to `/sys/ins/aclm.ins.pas` to enable proper subsystem ACL checking are documented in the *Domain/OS Call Reference*.) Following that is a shell script that installs the subsystem using the `crsubs`, `ensubs`, and `subs` commands.

Pascal Source Manager

The 'pse' program is used to extract the protected data from objects owned by the 'ps_example' subsystem and put them in an output file. As a trivial example, the protected data has a sequence number in the first 8 columns of each line, which is not logically part of the data, but which can be imagined to be important to the integrity of the data. Extracting the data removes the sequence number and copies the rest of the line to the output file. If this were a real application, it might also format and/or select the data sent to the output file.

```
{ pse --- protected subsystems example program }

{ usage:   pse pse_file out_file

  where:
    pse_file   protected object owned by 'ps_example' subsystem
    out_file   output file
}

program pse;

type
  buf_t = array[1..128] of char;

var
  istrid: stream_$id_t;      { input stream id }
  ostrid: stream_$id_t;      { output stream id }
  arg:    name_$pname_t;     { command line argument }
  alen:   integer;           { length of command line argument }
  st:     status_$t;         { status code }
  sk:     stream_$sk_t;      { stream seek key }
  buf:    buf_t;             { i/o buffer }
  bp:     ^buf_t;            { pointer to same }
  blen:   integer32;         { length of i/o buffer }

begin
  { get input file name }
  alen := pgm_$get_arg(1, arg, st, sizeof(arg));
  if (st.code <> 0) then begin
    writeln('input file name missing. ');
    error_$print(st);
    pgm_$set_severity(pgm_$error);
  end;
end;
```

```

    pgm_$exit
end;

{ open input file; must increase privilege to access
  my own protected file... }
aclm_$up;                                { get more privilege }
stream_$open(arg, alen, stream_$read,
  stream_$no_conc_write, istrld, st);
aclm_$down;                              { decrease privilege }
if (st.code <> 0) then begin
  writeln('Can''t open input file.');
```

```

  error_$print_name(st, arg, alen);
  pgm_$set_severity(pgm_$error);
  pgm_$exit
end;

{ get output file name }
alen := pgm_$get_arg(2, arg, st, sizeof(arg));
if (st.code <> 0) then begin
  writeln('output file name missing.');
```

```

  error_$print(st);
  pgm_$set_severity(pgm_$error);
  pgm_$exit
end;

{ create output file; DO NOT increase privilege: it would
  be an error to write on one of my own protected objects
  here -- I want an ordinary file }
stream_$create(arg, alen, stream_$overwrite,
  stream_$no_conc_write, ostrld, st);
if (st.code <> 0) then begin
  writeln('Can''t create output file.');
```

```

  error_$print_name(st, arg, alen);
  pgm_$set_severity(pgm_$error);
  pgm_$exit
end;

{ now just copy the file... a real program would be more
  complicated here. }
repeat
  { read a record... }
  aclm_$up;
  stream_$get_rec(istrld, addr(buf), 128, bp, blen, sk, st);
  aclm_$down;
```

```

if st.code <> 0 then          { error or EOF }
    exit;
{ write a record, stripping off the sequence number.
  Notice I did NOT make a check to see that the length
  of the record was greater than 8 characters... I am
  confident that the rest of the subsystem correctly
  maintains sequence numbers, and that the protected
  subsystem mechanism makes sure that only the subsystem
  can operate on the data. }
stream_$put_rec(ostrid, addr(bp^[9]), blen-8, sk, st);
until st.code <> 0;

{ check that we stopped because of EOF }
if (st.subsys = stream_$subs) and then
  (st.code = stream_$end_of_file) then
  pgm_$exit;

{ not EOF --- a real error of some sort, then }
writeln('i/o error: ');
error_$print(st);
pgm_$set_severity(pgm_$error);
pgm_$exit
end.

```

Shell Script

```

# create a protected subsystem
crsubs ps_example
# create some data to be protected --- normally a special create
# operation would be used that guarantees data integrity
catf >ps_data <<!
12345678this is some protected data
12345679next record of protected data
!
ensubs ps_example -v <<!
# enter the new subsystem in a shell
subs pse ps_example -mgr
# make pse a manager of 'ps_example'
subs ps_data ps_example -data
# protect the data
edacl ps_data -d % -a %.backup r
# now can only get at data from within

```

```

subs -up
cpf ps_data ps_data2 -subs
i           # make a copy of the data
subs -down
!
pse ps_data out_file
           # run pse to extract the protected data
catf out_file
i           # see the protected data
           # now see how it fails if I try
           #to make the output file a
           # protected object of the 'ps_example'
           #subsystem...
pse ps_data ps_data2
           # try to clobber ps_data2

```

SEE ALSO

More information is available. Type

help protection protected_subs

For a detailed description of protected subsystems

help crsubs

For details about creating a protected subsystem

help ensubs

For details about entering a protected subsystem

help xsubs

For details about executing a shell script as a subsystem manager

NAME

swedish_to_iso – convert files to ISO format

SYNOPSIS

swedish_to_iso *input_file output_file*

DESCRIPTION

These utilities convert files written with the overloaded 7-bit national fonts to the International Standards Organization (ISO) 8-bit format. The overloaded fonts include any with a specific language suffix (for example, `f7x13.french`, or `din_f7x11.german`). If you created and/or edited a file using one of the national fonts, that file is a candidate for conversion.

You are not required to convert files, but probably will want to because

1. The overloaded fonts replace certain ASCII characters with national ones, have that subset of ASCII characters and the national characters in one file. The 8-bit fonts available as of SR10 include all the ASCII characters and the national characters.
2. The 8-bit fonts also include a wider range of national characters, so you can enter any character in any western European language. This was not always possible with the overloaded fonts. For example, there was not enough space in the overloaded font to include all the French characters, but they all exist in the 8-bit fonts.

There are two parameters to the conversion utilities. You must provide the name of the file you want to convert (*input_file*) and your *output_file*. If *output_file* already exists, the utilities abort.

The default location for the utilities is `/usr/apollo/bin`.

FILES

<code>/usr/apollo/bin/french_to_iso</code>	Converts overloaded French to ISO format
<code>/usr/apollo/bin/german_to_iso</code>	Converts overloaded German to ISO format
<code>/usr/apollo/bin/nor.dan_to_iso</code>	Converts overloaded Norwegian/Danish to ISO format
<code>/usr/apollo/bin/swedish_to_iso</code>	Converts overloaded Swedish/Finnish to ISO format
<code>/usr/apollo/bin/swiss_to_iso</code>	Converts overloaded Swiss to ISO format
<code>/usr/apollo/bin/uk_to_iso</code>	Converts overloaded U.K. English to ISO format

DIAGNOSTICS

All messages are generally self-explanatory.

NAME

`swiss_to_iso` – convert files to ISO format

SYNOPSIS

`swiss_to_iso` *input_file* *output_file*

DESCRIPTION

These utilities convert files written with the overloaded 7-bit national fonts to the International Standards Organization (ISO) 8-bit format. The overloaded fonts include any with a specific language suffix (for example, `f7x13.french`, or `din_f7x11.german`). If you created and/or edited a file using one of the national fonts, that file is a candidate for conversion.

You are not required to convert files, but probably will want to because

1. The overloaded fonts replace certain ASCII characters with national ones, have that subset of ASCII characters and the national characters in one file. The 8-bit fonts available as of SR10 include all the ASCII characters and the national characters.
2. The 8-bit fonts also include a wider range of national characters, so you can enter any character in any western European language. This was not always possible with the overloaded fonts. For example, there was not enough space in the overloaded font to include all the French characters, but they all exist in the 8-bit fonts.

There are two parameters to the conversion utilities. You must provide the name of the file you want to convert (*input_file*) and your *output_file*. If *output_file* already exists, the utilities abort.

The default location for the utilities is `/usr/apollo/bin`.

FILES

<code>/usr/apollo/bin/french_to_iso</code>	Converts overloaded French to ISO format
<code>/usr/apollo/bin/german_to_iso</code>	Converts overloaded German to ISO format
<code>/usr/apollo/bin/nor.dan_to_iso</code>	Converts overloaded Norwegian/Danish to ISO format
<code>/usr/apollo/bin/swedish_to_iso</code>	Converts overloaded Swedish/Finnish to ISO format
<code>/usr/apollo/bin/swiss_to_iso</code>	Converts overloaded Swiss to ISO format
<code>/usr/apollo/bin/uk_to_iso</code>	Converts overloaded U.K. English to ISO format

DIAGNOSTICS

All messages are generally self-explanatory.

NAME

`syncids` – fix or verify file owners in a file system

SYNOPSIS

`/etc/syncids [-a] [-l] [-v] volume-pathname`

DESCRIPTION

In a Domain system, every user (and group) is identified by a 64-bit identifier that is unique across all Domain users (and groups) in both space and time. However, UNIX interfaces that take user and group IDs as arguments expect integers that may be unique only within a given file system. The Domain file system stores both forms of identifier with each file. The 64-bit UIDs are used for checking access rights, since there can never be conflicts even if two networks are merged, or a workstation moves from one network to another.

`syncids` is a program that should be run whenever network registries are merged, or a node is moved between networks having different registries. It ensures that the UNIX owner IDs match the unique owner IDs for every object on the logical volume.

OPTIONS

- `-a` List the name and owner information for each object as it is processed.
- `-l` Only list information about objects for which UNIX owner IDs are incorrect.
- `-v` Verify only; do not actually modify the owners of any objects. This option implies `-l`. If the `-a` option is also given, then information will be printed about every object on the volume.

NAME

tb – print process traceback

SYNOPSIS

tb [*options*] [*process_spec*]

DESCRIPTION

tb prints a process traceback, listing the name and current line number of each routine on the call stack. There are two forms of traceback:

- Active Traces the current state of an executing process.
- Diagnostic Traces the state of an aborted process at the time of the fault which killed it.

Note:

There is a homonymous DM command: **tb** (to bottom of window). Type **help tb_dm** for details about that command.

process_spec (optional)

UNIX process ID (PID), aegis process name, or aegis process UID. Process names are not recorded in the process dump file, so dead processes must be referenced by PID or UID. Since PID's are reused multiple dump file entries for the same PID are possible, the command will select the most recent.

Default if omitted: perform a diagnostic traceback for the last child of the invoking process

OPTIONS

- p[roc]** Trace exactly the specified process. If this option is absent, the specified process or one of its children may be traced, as described below.
- d[iagnostic]** Print a diagnostic traceback of an aborted process.
- n[ode] node_spec** Use the process dump file on the specified node. Implies **-diagnostic**.
- c[ommand] pathname** Print diagnostic traceback(s) for processes running the specified program. *pathname* must be reachable from the working directory; command search rules are not applied. Implies **-diagnostic**.
- s[ince] date_time_spec** Print diagnostic traceback(s) for processes which aborted after the specified time. Implies **-diagnostic**. The format for *date_time_spec* is [[*yyyy/mm/dd*][.*hh:mm[:ss]*]].

- l[ast] [n]** Print the *n* most recent entries in the process dump file (which also satisfy other selection criteria if given). *n* defaults to 1. If neither **-last** nor **-all** is specified **tb** prints only the most recent entry. Implies **-diagnostic**.
- a[ll]** Print all entries in the process dump file (which also satisfy other selection criteria if given.) If neither **-last** or **-all** are specified, **tb** prints only the most recent entry. Implies **-diagnostic**.
- f[ull]** Print additional fault diagnostic information, such as register values. Implies **-diagnostic**.
- b[rief]** List entries in the process dump file that satisfy selection criteria, but do not print tracebacks. The listing shows the process, parent, and group IDs, the time of the dump, the abort status, and the program that was running.
- t[asks]** Trace all tasks in the process. By default only the currently active task is shown. Ignored if tasking is not enabled. Applies only to active process tracebacks.
- h[eaders_off]** Suppresses output of process ID, dump time, and program name preceding diagnostic traceback, or of column headers in brief format. It has no effect on active process traceback.

Diagnostic Tracebacks

A diagnostic traceback shows the state of the call stack at the time of a fault which causes a process to be aborted. Traceback information is written to `'node_data/system_logs/proc_dump` at the time of the fault. This is a circular buffer in which the oldest information is overwritten as needed to make room for new. There is space for approximately 150-200 dumps. **tb** prints up to 128 call levels for diagnostic tracebacks.

tb prints a diagnostic traceback if the command line specifies **-diagnostic** or any option which implies it, or if the process specified is not active. If **-diagnostic** is specified together with an active process, the most recent aborted child of that process is traced (or most recent children if **-last** or **-all** is specified).

If no options are given (except possibly **-f**, **-b** or **-h**) **tb** prints a diagnostic traceback for the most recent aborted child of the process which invoked **tb**.

Examples of Requesting Diagnostic Tracebacks

Assume `process_5` is an active shell process, and process number 107 is not active. Traceback process 107.

```
$ tb 107
```

Traceback last aborted command invoked from `process_5`.

```
$tb -d process_5
```

Traceback last aborted command from this shell

```
$tb
```

Traceback last aborted process running `test3`

```
$tb -c test3
```

List all entries in the process dump file made today

```
$tb -s today -a -b
```

Active Process Tracebacks

An active process traceback shows the current state of an executing process, listing the name and line number of each procedure in the call stack. The process is suspended while the traceback is taken. `tb` prints an active process traceback if the command line specifies an active process and does not include `-diagnostic` (or any option that implies it). If the process is specified by name and has any active children, then the most recent child is traced. (This allows a process to be specified by the name of its invoking shell process.) This behavior may be overridden by the `-proc` switch, or by specifying the process by PID or UID. Note that the only other option applicable to active process tracebacks is `-task`.

Examples of Requesting Active Process Tracebacks

Assume `process_7` is an active shell process, from which a command running in process 747 has been invoked.

```
$tb 747
```

Traceback the invoked command

```
$tb process_7
```

same

```
$tb -p process_7
```

Traceback the shell process itself

NAME

`tcpstat` – show network status

SYNOPSIS

```
tcpstat [ -Aang ] [ -f address_family ]
tcpstat [ -himnrstT ] [ -f address_family ]
tcpstat [ -n ] [ -I interface ] interval
```

DESCRIPTION

The `tcpstat` command symbolically displays the contents of various network-related data structures. You can specify one of a number of output formats. The first form of the command displays a list of active sockets for each protocol. The second form presents the contents of one of the other network data structures according to the option selected. The third form, with an *interval* specified, continuously displays the information regarding packet traffic on the configured network interfaces.

The default display, for active sockets, shows the local and remote addresses, send and receive queue sizes (in bytes), protocol, and the internal state of the protocol. Address formats are of the form *host.port* or *network.port* if a socket's address specifies a network but no specific host address. It displays the host and network addresses, when known, symbolically, according to the databases `/etc/hosts` and `/etc/networks`, respectively. If a symbolic name for an address is unknown, or if you specify the `-n` option, `tcpstat` displays the address numerically, according to the address family. `tcpstat` displays unspecified, or "wildcard", addresses and ports an asterisk (*).

The interface display provides a table of cumulative statistics regarding packets transferred, errors, and collisions. It also shows the network addresses of the interface and the maximum transmission unit (mtu).

The routing table display indicates the available routes and their status. Each route consists of a destination host or network and a gateway to use in forwarding packets.

The flags field shows the following:

- The state of the route (U if "up")
- Whether the route is to a gateway (G)
- Whether the route was created dynamically by a redirect (D)
- Whether the route has priority (P)
- Whether the route is a static (S) route added with `route`
- Whether the route has been marked for deletion (X).

Direct routes are created for each interface attached to the local host; the gateway field for such entries shows the address of the outgoing interface. The `refcnt` field gives the current number of active uses of the route. Connection oriented protocols normally hold on to a single route for the duration of a connection while connectionless protocols obtain a route while sending to the same destination. The `use` field provides a count of

the number of packets sent using that route. The interface entry indicates the network interface utilized for the route.

When you invoke `tcpstat` with an *interval* argument, it displays a running count of statistics related to network interfaces. This display consists of a column for the primary interface (the first interface found during auto-configuration) and a column summarizing information for all interfaces. Use the `-I` option to replace the primary interface with another interface. The first line of each screen of information contains a summary since the system was last rebooted. Subsequent output lines show values accumulated over the preceding interval.

OPTIONS

- `-A` With the default display, show the address of any protocol-control blocks associated with sockets; used for debugging.
- `-a` With the default display, show the state of all sockets; normally sockets used by server processes are not shown.
- `-g` With the default display, show the first gateway used.
- `-h` Show the state of the IMP host table. Status flags show the gateway entry (G), in use (U), a temporary entry (T).
- `-i` Show the state of interfaces that were auto-configured (`tcpstat` does not show interfaces statically configured into a system, but not located at boot time).
- `-I interface` Show information only about this interface; used with an *interval* as described below.
- `-m` Show statistics recorded by the memory-management routines (the network manages a private pool of memory buffers).
- `-n` Show network addresses as numbers (normally `tcpstat` interprets addresses and attempts to display them symbolically). You can use this option with any of the display formats.
- `-s` Show per-protocol and routing statistics.
- `-r` Show the routing tables.
- `-t` When used with the `-i` option, show timer column.
- `-T` Show all possible status information.
- `-f address_family` Limit statistics or address-control-block reports to those of the specified *address family*. The following address families are recognized: `inet`, for `AF_INET`; `ns`, for `AF_NS`; and `unix`, for `AF_UNIX`.

CAUTIONS

The notion of errors is ill-defined. Collisions mean something else for the IMP.

NAME

tctl – set or display SIO line characteristics

SYNOPSIS

tctl [*arguments*] [*options*]

DESCRIPTION

tctl sets or displays SIO line characteristics, which control how hardware and software connected to those lines should behave. For example, if you wish to allow a dumb terminal to dial into a node and communicate meaningfully with a shell, you must properly configure the SIO line that the terminal can use so that the node will understand the terminal's signals. Thus, **tctl** controls the transmission speed (baud rate) that connected terminals must use, and which characters typed on those terminals delete characters or lines.

COMMAND LINE SUMMARY

-line number	Set the line number to which this configuration applies.
-nld [msec]	Set newline delay, in milliseconds. The default is 20.
-speed rate	set baud rate.
-force	Set baud rate even if it affects partner line.
-erase chr	Set erase character.
-kill chr	Set kill character.
-eof chr	Set end-of-file character.
-quitchr chr	Set quit character.
-intchr chr	Set interrupt character (used primarily by Domain/OS).
-suspchr chr	Set suspend character (used primarily by Domain/OS).
-parity none	Don't send or check parity bit.
 even	Send and check even parity.
 odd	Send and check odd parity.
-bpc {5 6 7 8}	Set number of bits per character.
-stop {1 1.5 2}	Set number of stop bits.
-[no]raw	Enable/disable "raw" mode.
-[no]echo	Enable/disable echo.
-[no]sync	Enable/disable host synchronization via node sending CTRL-S/CTRL-Q (enable implies -norts_enable).
-[no]insync	Enable/disable input synchronization, honoring CTRL-S/CTRL-Q sequences received.

- [no]cvt_nl** Enable/disable transmitting new_lines (NL=10) as CR-LF (when using the **emt** command, this option must be considered in conjunction with the setting of the **outterm** state in **emt**).
- [no]cvtraw_nl** Enable or disable transmitting new_lines (NL=10) as CR-LF in "raw" mode.
- [no]quit** Enable/disable passing quits received to this process.
- [no]int** Enable/disable passing interrupt faults to this process.

- [no]susp** Enable/disable passing suspension faults to this process.
- [no]rts** Set/reset the request-to-send line.
- [no]dtr** Set/reset the data-terminal-ready line.
- [no]dcd_enable** Enable/disable standard handling of carrier detect.
- [no]cts_enable** Enable/disable standard handling of clear-to-send.
- [no]rts_enable** Enable/disable synchronization via the request-to-send line (enable implies **-nosync**).
- [no]bp_enable** Enable/disable processing of bitpad input on this line.
- error [no]framing** Enable/disable report of framing errors.
- [no]parity** Enable/disable report of parity errors.
- [no]dcd_change** Enable/disable report of change in DCD line.
- [no]cts_change** Enable/disable report of change in CTS line.
- default** Set all settable options to default values.

Valid speeds are 50, 75, 110, 134, 150, 300, 600, 1200, 2000, 2400, 3600, 4800, 7200, 9600, and 19200. **chr** may be a single ASCII character or a one-byte hexadecimal value (for example, **1a** or **0f1**).

OPTIONS

If no options are specified, the current settings of the SIO lines are displayed.

- default** Set all settable options to their default values. This allows you to quickly reset values to known states.
- line *n*** Specify the SIO line to be affected by subsequent options on this command line. *n* is an integer in the range 0-3. The default SIO line is line 1 or standard input (if standard input is directed to an SIO line).

- speed *baud*** Set the speed of the line, for both input and output. The possible baud rates are: 50, 75, 110, 134, 150, 300, 600, 1200, 2000, 2400, 3600, 4800, 7200, 9600, 19200. The initial setting is 9600 baud. Note that 3600 baud is not supported on DN3xx systems. Speeds for partner line(s) may occasionally need to be forced: see **-force** below.
- force** Valid only if **-speed** is also specified. This option forces the speed of the line specified by **-line** to be set to the correct speed (specified by **-speed**). If the line has a partner line that is currently set to some other (incompatible) speed, **-force** will reset the partner line's speed to 9600 baud. See example 4 below. For more information about partner lines, see the `sio_$control` description in *Domain/OS Call Reference*.
- nld [*n*]** Set newline delay. This is the number of milliseconds required following the output of a line feed (newline). If *n* is omitted or not set, 20 milliseconds is the default.
- erase *char*** Set the erase character. This option is valid only when data is being passed to the SIO line in "cooked" mode. *char* may be any character or a one-byte hexadecimal value. Some characters may require quoting in the shell. The erase character is initially set to BACKSPACE (08 hex).
- kill *char*** Set the kill character. This option is valid only when data is being passed to the SIO line in "cooked" mode. The kill character is initially set to CTRL/X.
- eof *char*** Set the end-of file character. The EOF character is initially set to CTRL/Z.
- quitchr *char*** Set the quit character. The quit character is initially set to CTRL/].
- intchr *char*** Set the interrupt character. This is used primarily by Domain/OS. The interrupt character is initially CTRL/C.
- suspendchr *char*** Set the suspend character. This is used primarily by Domain/OS. The suspend character is initially CTRL/P.
- [no]raw** Turn "raw" mode on or off. In "raw" mode, full 8-bit bytes are transmitted in both directions, without any interpretation. The initial setting is **-noraw**.
- [no]echo** Turn the echoing of input characters over the SIO line on or off. The initial setting is **echo**.
- [no]sync** A standard terminal protocol for synchronization is the sending of CTRL/S (XON) when the terminal input buffer begins to fill, and CTRL/Q (XOFF) when the buffer begins to empty. This protocol is used to control the flow of transmissions from a high-speed data source (when the node is receiving information too fast from a host).

- This option enables or disables this synchronization behavior (it is initially enabled). `-sync` implies `-norts_enable`.
- `-[no]cvt_nl` Enable or disable conversion of LF to CR-LF on output. `cvt_nl` causes newline (LF) to be transmitted as CR-LF sequences. This option is valid only when data is being passed to the SIO line in "cooked" mode. The initial setting is `-nocvt_nl`.
- `-[no]cvtraw_nl` Similar to `-cvt_nl`, but applies only to "raw" mode.
- `-[no]insync` When a node is transmitting data on a serial line, the terminal (or host on the receiving end) may not be able to keep up with the node transmissions and sends CTRL/S to stop the node from transmitting, then CTRL/Q to resume. This option is used enable or disable reacting to CTRL/S and CTRL/Q when they are received by the node. `-insync` causes transmissions to halt when CTRL/S is received and to resume when CTRL/Q is received. The initial setting is `-noinsync`.
- `-parity state` Select parity checking state. Valid states are as follows:
- | | |
|-------------------|---|
| <code>none</code> | Don't send or check parity bit. |
| <code>even</code> | Send and check even parity. |
| <code>odd</code> | Send and check odd parity. The initial state is none. |
- `-bpc n` Set number of bits per character. `n` is an integer in the range 5-8. The initial number of bits per character is 8.
- `-stop n` Set number of stop bits. `n` may be 1, 1.5, or 2. The initial number of stop bits is 1.
- `-[no]quit` Enable/disable quits for the current process. The initial setting is `-noquit`.
- `-[no]int` Enable/disable interrupts for the current process. The initial setting is `-noint`.
- `-[no]susp` Enable/disable suspend faults for the current process. The initial setting is `-nosusp`.
- `-[no]rts` Enable/disable the request-to-send line. The initial setting is `-rts`. Note that you may not use this option if `-rts_enable` is specified.
- `-[no]dtr` Enable/disable the data-terminal-ready line. The initial setting is `-dtr`. Note that `-dtr` is not valid if `-line 3` is specified.
- `-[no]dcd_enable` Enable/disable standard handling of carrier detect. The initial setting is `-nodcd_enable`.
- `-[no]cts_enable` Enable/disable standard handling of clear-to-send. The initial setting is `-nocts_enable`.

- [no]rts_enable** Enable/disable RTS flow control. The initial setting is **-norts_enable**. Enable implies **-nosync**.
- [no]bp_enable** Enable/disable processing of bit-pad input (from a graphics tablet) on the SIO line. When enabled, data received on this line is not delivered through **ios_\$get**, but is accumulated by the interrupt routine, and passed to the display driver a point at a time, much as with the touchpad. This processing has the additional property that subsequent points within +/-1 in both the x and y dimensions are ignored. The initial setting is **-nobp_enable**.
- error state** Select error reporting state. Valid states are as follows:
- | | |
|-----------------------|---|
| [no]framing | Enable/disable reported framing errors. |
| [no]parity | Enable/disable reported parity errors. |
| [no]dcd_change | Enable/disable report on DCD line. |
| [no]cts_change | Enable/disable report on CTS line. |

Only framing is initially enabled.

NOTE

emt always puts the SIO line in "raw" mode, so **-cvt_nl** has no effect in that instance. Use the **outterm** command within **emt**.

EXAMPLES

```
$ tctl      Display current settings.
Status of Line 1:
Erase (character delete) character: 08 (hex)
Kill (line delete) character: 18 (hex)
End of file character: 1A (hex)
Quit character: 1D (hex)
Interrupt character: 03 (hex)
Suspend character: 10 (hex)
New line delay: 0
Speed: 9600
Raw: FALSE,          Echo: TRUE,          Cvt_NL: TRUE
CvtRaw_NL: FALSE,   Host_Sync: TRUE,    Input_Sync: FALSE
RTS: TRUE,          DTR: TRUE,         DCD: FALSE
CTS: FALSE,        Quit_Enable: FALSE, Int_Enable: FALSE
Susp_Enable: FALSE, DCD_Enable: FALSE, CTS_enable: FALSE
BP_enable: FALSE   RTS_enable: FALSE
Eight bits per character, Parity: None, One stop bit
Errors enabled: FRAMING
```

Set quit character to hex FE, enable input synchronization, set speed to 300 baud on SIO line 2.

```
$ tctl -line 2 -quitchar 0FE -insync -speed 300
```

Set parity to odd, quit character to # (quoted because # normally begins a comment in the shell), and kill character to ! on line 1.

```
$ tctl -parity odd -quitchar '#' -kill !
```

```
$ tctl -line 2 -speed 50
```

```
?(tctl) Speed requested is incompatible with current
speed of partner line 1. Resubmit command
with -force if permissible to reset partner
line to 9600 baud.
```

```
$ tctl -line 2 -speed 50 -force
```

```
?(tctl) Warning: Speed of partner line has been reset
to 9600 baud.
```

SEE ALSO

More information is available. Type

help emt

For details about configuring your node as a dumb terminal to communicate with a remote host via an SIO line

NAME

tee – copy input to output and to named files

SYNOPSIS

tee *pathname* ...

DESCRIPTION

tee copies its standard input to standard output and to the named files. It is useful for saving the data being transmitted through a pipeline.

ARGUMENTS

pathname (required) Specify name of file to receive output. Multiple pathnames are permitted.

EXAMPLES

```
$ fmt mary | tee mary.clean | os >mary.overstruck
```

This command line causes the file `mary` to be formatted with `fmt`. The formatted text is written to the file `mary.clean` and also piped to the `os` command to produce overstruck output (for a line printer) redirected into the file `mary.overstruck`. Thus, you end up with two output files: one with ASCII carriage control `mary.clean` and one with FORTRAN carriage control `mary.overstruck`.

NAME

telnet – user interface to the TELNET protocol

SYNOPSIS

telnet [*host* [*port*]]

DESCRIPTION

telnet is used to communicate with another host using the TELNET protocol. If **telnet** is invoked without arguments, it enters command mode, indicated by its prompt (“telnet>”). In this mode, it accepts and executes the commands listed below. If it is invoked with arguments, it performs an open command (see below) with those arguments.

Once a connection has been opened, **telnet** enters an input mode. The input mode entered will be either “character at a time” or “line by line” depending on what the remote system supports.

In “character at a time” mode, most text typed is immediately sent to the remote host for processing.

In “line by line” mode, all text is echoed locally, and (normally) only completed lines are sent to the remote host. The “local echo character” (initially “E”) may be used to turn off and on the local echo (this would mostly be used to enter passwords without the password being echoed).

In either mode, if the **localchars** toggle is TRUE (the default in line mode; see below), the user’s **quit**, **intr**, and **flush** characters are trapped locally, and sent as TELNET protocol sequences to the remote side. There are options (see **toggle autoflush** and **toggle autosynch** below) which cause this action to flush subsequent output to the terminal (until the remote host acknowledges the TELNET sequence) and flush previous terminal input (in the case of **quit** and **intr**).

While connected to a remote host, **telnet** command mode may be entered by typing the **telnet** “escape character” (initially “^”). When in command mode, the normal terminal editing conventions are available.

COMMANDS

The following commands are available. Only enough of each command to uniquely identify it need be typed (this is also true for arguments to the **mode**, **set**, **toggle**, and **display** commands).

open *host* [*port*]

Open a connection to the named host. If no port number is specified, **telnet** will attempt to contact a TELNET server at the default port. The host specification may be either a host name or an Internet address specified in “dot notation”.

close

Close a TELNET session and return to command mode.

- quit** Close any open TELNET session and exit telnet. An end of file (in command mode) will also close a session and exit.
- z** Suspend telnet. This command only works when the user is using the `cs`.
- mode type** Ask the remote host for permission to go into the requested mode. If the remote host is capable of entering that mode, the requested mode will be entered. *Type* is either line (for "line by line" mode) or character (for "character at a time" mode).
- status** Show the current status of telnet. This includes the peer one is connected to, as well as the current mode.
- display [argument...]**
Displays all, or some, of the set and toggle values (see below).
- ? [command]**
Get help. With no arguments, telnet prints a help summary. If a command is specified, telnet prints the help information for just that command.
- send arguments**
Sends one or more special character sequences to the remote host. The following are the arguments which may be specified (more than one argument may be specified at a time):
- escape** Sends the current telnet escape character (initially "\^").
 - synch** Sends the TELNET SYNCH sequence. This sequence causes the remote system to discard all previously typed (but not yet read) input. This sequence is sent as TCP urgent data (and may not work if the remote system is a 4.2 BSD system — if it doesn't work, a lower case "r" may be echoed on the terminal).
 - brk** Sends the TELNET BRK (Break) sequence, which may have significance to the remote system.
 - ip** Sends the TELNET IP (Interrupt Process) sequence, which should cause the remote system to abort the currently running process.
 - ao** Sends the TELNET AO (Abort Output) sequence, which should cause the remote system to flush all output from the remote system to the user's terminal.
 - ayt** Sends the TELNET AYT (Are You There) sequence, to which the remote system may or may not choose to respond.
 - ec** Sends the TELNET EC (Erase Character) sequence, which should cause the remote system to erase the last character entered.

- el** Sends the TELNET EL (Erase Line) sequence, which should cause the remote system to erase the line currently being entered.
- ga** Sends the TELNET GA (Go Ahead) sequence, which likely has no significance to the remote system.
- nop** Sends the TELNET NOP (No OPeration) sequence.
- ?** Prints out help information for the send command.

set argument value

Set any one of a number of telnet variables to a specific value. The special value **off** turns off the function associated with the variable. The values of variables may be interrogated with the **display** command. The variables which may be specified are:

- echo** This is the value (initially “‘E’”) which, when in “line by line” mode, toggles between doing local echoing of entered characters (for normal processing), and suppressing echoing of entered characters (for entering, say, a password).
- escape** This is the telnet escape character (initially “‘[’”) which causes entry into telnet command mode (when connected to a remote system).

interrupt

If telnet is in **localchars** mode (see **toggle localchars** below) and the **interrupt** character is typed, a TELNET IP sequence (see **send ip** above) is sent to the remote host. The initial value for the interrupt character is taken to be the terminal’s **intr** character.

- quit** If telnet is in **localchars** mode (see **toggle localchars** below) and the **quit** character is typed, a TELNET BRK sequence (see **send brk** above) is sent to the remote host. The initial value for the quit character is taken to be the terminal’s **quit** character.

flushoutput

If telnet is in **localchars** mode (see **toggle localchars** below) and the **flushoutput** character is typed, a TELNET AO sequence (see **send ao** above) is sent to the remote host. The initial value for the flush character is taken to be the terminal’s **flush** character.

- erase** If telnet is in **localchars** mode (see **toggle localchars** below), and if telnet is operating in “character at a time” mode, then when this character is typed, a TELNET EC sequence (see **send ec** above) is sent to the remote system. The initial value for the erase character is taken to be the terminal’s **erase**

character.

- kill** If telnet is in **localchars** mode (see **toggle localchars** below), and if telnet is operating in "character at a time" mode, then when this character is typed, a TELNET EL sequence (see **send el** above) is sent to the remote system. The initial value for the kill character is taken to be the terminal's kill character.
- eof** If telnet is operating in "line by line" mode, entering this character as the first character on a line will cause this character to be sent to the remote system. The initial value of the eof character is taken to be the terminal's eof character.

toggle arguments...

Toggle (between TRUE and FALSE) various flags that control how telnet responds to events. More than one argument may be specified. The state of these flags may be interrogated with the **display** command. Valid arguments are:

localchars

If this is TRUE, then the **flush**, **interrupt**, **quit**, **erase**, and **kill** characters (see **set** above) are recognized locally, and transformed into (hopefully) appropriate TELNET control sequences (respectively **ao**, **ip**, **brk**, **ec**, and **el**; see **send** above). The initial value for this toggle is TRUE in "line by line" mode, and FALSE in "character at a time" mode.

autoflush

If **autoflush** and **localchars** are both TRUE, then when the **ao**, **intr**, or **quit** characters are recognized (and transformed into TELNET sequences; see **set** above for details), telnet refuses to display any data on the user's terminal until the remote system acknowledges (via a TELNET *Timing Mark* option) that it has processed those TELNET sequences. The initial value for this toggle is TRUE.

autosynch

If **autosynch** and **localchars** are both TRUE, then when either the **intr** or **quit** characters is typed (see **set** above for descriptions of the **intr** and **quit** characters), the resulting TELNET sequence sent is followed by the TELNET SYNCH sequence. This procedure should cause the remote system to begin throwing away all previously typed input until both of the TELNET sequences have been read and acted upon. The initial value of this toggle is FALSE.

- crmod** Toggle carriage return mode. When this mode is enabled, most carriage return characters received from the remote host will be

mapped into a carriage return followed by a line feed. This mode does not affect those characters typed by the user, only those received from the remote host. This mode is not very useful unless the remote host only sends carriage return, but never line feed. The initial value for this toggle is FALSE.

debug Toggles socket level debugging (useful only to the super-user). The initial value for this toggle is FALSE.

options Toggles the display of some internal telnet protocol processing (having to do with TELNET options). The initial value for this toggle is FALSE.

netdata Toggles the display of all network data (in hexadecimal format). The initial value for this toggle is FALSE.

? Displays the legal toggle commands.

CAUTIONS

There is no adequate way for dealing with flow control.

On some remote systems, echo has to be turned off manually when in "line by line" mode.

There is enough settable state to justify a .telnetrc file.

No capability for a .telnetrc file is provided.

In "line by line" mode, the terminal's eof character is only recognized (and sent to the remote system) when it is the first character on a line.

NAME

tlc – replace characters

SYNOPSIS

tlc *from-chars* [*to-chars*]

DESCRIPTION

tlc copies standard input to standard output, substituting or deleting selected characters. Each input character found in *from-chars* is replaced by the corresponding character of *to-chars*.

tlc differs from **chpat** (*change_pattern*) in that it deals only with single characters or ranges of characters, whereas **chpat** deals with character strings. For example,

\$ tlc xy yx

changes all x's into y's and all y's into x's, whereas

\$ chpat xy yx

changes all the patterns xy into yx.

ARGUMENTS

from-chars (required) Specify existing character(s) to be replaced. You may specify a range of characters by separating the extremes with a dash. For example, a-z stands for the list of lowercase letters. *from-chars* may contain a maximum of 100 characters.

to-chars (optional) Specify replacement characters. You may specify a range of characters by separating the extremes with a dash. For example, a-z stands for the list of lowercase letters. *to-chars* may contain a maximum of 100 characters.

If *from-chars* and *to-chars* contain an equal number of characters, **tlc** translates the first character in *from-chars* to the first character in *to-chars*, and so forth.

If *from-chars* contains more characters than *to-chars*, **tlc** repeats the last character in *to-chars* until *to-chars* is as long as *from-chars*. However, in the output, adjacent repetitions of the last character appear as one character. (See example 2 below.)

If *to-chars* contains more characters than *from-chars*, the extra characters are ignored.

Default if omitted: delete all occurrences of characters in the *from-chars* list

EXAMPLES

The following examples show `tlc`'s operation using standard input and output. The first line following the command line is an echo of standard input. The next line is the `tlc` results, then another line of input, then more results, and so forth.

1.

```
$ tlc te zq
Now is the time
Now is zhq zimq
*** EOF ***
$
```

2.

```
$ tlc abc zq
Now is the time for all good men and boys to come to the aid
Now is the time for zll good men znd qoys to qome to the zid
abcacbaa
zqzqzz
aaaaa
zzzzz
bbbbbb
q
ccccc
q
*** EOF ***
```

Note that multiple occurrences of `a` are replaced by `z` one for one, but multiple occurrences of `b` and `c` are replaced with a single `q`, since the *from-char* list is longer than the *to-char* list.

3.

```
$ tlc A-Z a-z <mary.caps >mary.lc
```

This command changes all uppercase letters in the input file `mary.caps` to lowercase and writes the results to the file `mary.lc`. Lowercase characters already in `mary.caps` remain unchanged.

NAME

tpm – set/display touchpad and mouse characteristics

SYNOPSIS

tpm [*options*]

DESCRIPTION

tpm allows you to define characteristics for the touchpad and mouse. The touchpad operates in one of three modes: absolute, relative, and absolute/relative. The mode of operation establishes how movements of your finger on the touchpad affect the position of the cursor on the screen. The three modes differ primarily in how the cursor moves when you lift your finger from the touchpad and then replace it. The mouse operates in relative mode only, and **-s** is the only valid option.

The subsections below describe the three operational modes, as well as the other options.

OPTIONS

If no options are specified, **tpm** displays the current touchpad characteristics.

- a** (default) Select absolute mode.
- r** Select relative mode.
- ar** Select absolute/relative mode.
- rerange** Set prescaling factors for touchpad data.
- s x y** Set scaling factors for *x* and *y*. Values can range from -32768 to 32767 . The default scaling factors are 799 for *x* and 1023 for *y* (portrait displays); and 1023 for *x* and 799 for *y* (landscape displays).
- o x y** Set *x* and *y* as the origin for absolute mode. Values must be in raster units, and can range from 0 to 1023. The default origin is 0,0.
- h n** Set the hysteresis box size. The value must be in raster units, and can range from 0 to 1023. The default is 5.

DESCRIPTION**Absolute Mode**

In absolute mode, using the default scale and origin, the touchpad approximates the screen, so that the top left edge of the touchpad represents cursor positions at the top left edge of the screen. Absolute mode is the default setting. When you place your finger on the touchpad, the cursor jumps to a corresponding position on the screen. Moving your finger across the touchpad moves the cursor across the screen in the same direction.

For example, moving your finger from the top of the touchpad to the bottom moves the cursor from top to bottom on the screen. If you lift your finger from the touchpad, and later touch the pad again, the cursor jumps to a new position on the screen corresponding to the new finger position.

Relative Mode

In relative mode, cursor movements correspond only to finger movements across the touchpad. The cursor does not move when you first place your finger on the touchpad. This differs from absolute mode, where the cursor jumps to a new position when you lift your finger and then replace it. In effect, relative mode causes the touchpad to correspond to different areas of the screen, relative to the current cursor position.

This is the only meaningful mode for a mouse: all movement begins from the current cursor position.

Relative mode is typically used with scale factors less than the defaults. Smaller scale factors mean that the touchpad maps to a smaller area of the screen. For example, scale factors of 200 by 256 specify one-sixteenth of a portrait display's screen area. With small scale factors, relative mode allows fine resolution of the cursor position within a small area.

To reach distant areas on the screen, you can use several strokes on the touchpad or mouse, each stroke moving the cursor closer to its final destination. To assist you in making large movements in relative mode without having to use too many strokes, the speed of cursor movement is artificially accelerated in relation to the speed of finger or mouse movement. Thus, a quick motion moves the cursor farther than a slow, deliberate motion which covers the same distance.

Absolute/Relative Mode

Absolute/relative mode is a combination of absolute and relative modes. It has no meaning for the mouse. In this mode, the first position of your finger on the touchpad establishes the first position of the cursor, as in absolute mode. Moving your finger across the touchpad moves the cursor across the screen. As in relative mode, the scale is typically smaller than the whole screen.

Unlike absolute and relative modes, however, the effect of lifting your finger from the touchpad depends on how long you break contact. If you lift and replace your finger quickly — within a half second — the cursor does not move, and the effect is the same as relative mode. If you break contact for more than a half second, however, the cursor jumps to a new absolute position when you put your finger on the touchpad again.

Absolute/relative mode is useful for jumping the cursor from one place to another, then carefully positioning it in the new area. For example, this mode is commonly used to move the cursor in a jump from one window to another, and then point to a character in the second window.

Prescaling the Touchpad

Raw touchpad data varies slightly from one touchpad to another. Prescaling is, in essence, calibration of the touchpad. Every time you start the node, the touchpad manager prescales the data to determine an exact range for the device.

To prescale, the touchpad manager observes the first thousand points of touchpad data (about 30 seconds of use). During this time, you should try to touch all four edges of the touchpad to ensure that the observed data constitutes an accurate sample. Based on the observed data, the touchpad manager computes a prescaling factor which, when applied to the data, brings all points into the range from -0.05 to 1.05 . This range corresponds to the edges of the screen, plus an overlap of 5%, when multiplied by the default scaling factors. Because of the overlap, you need not touch the internal frame (under the conductive material) to move the cursor to the edge of the screen.

The `-r` option invokes prescaling. This option is useful if the first 30 seconds of use did not include the entire range of the touchpad. It is also handy if you change keyboards on a node, and therefore need to reset the prescaling factors without restarting the node.

Scale Factors

The touchpad manager translates, or scales, the data into raster units, which the Display Manager understands. Scale factors, specified with the `-s` option, are applied to the prescaled touchpad data to translate it to raster units for the Display Manager.

The scale factors are multiplied by the prescaled data. The default scale factors are 800 for x and 1024 for y (portrait displays); and 1024 for x and 800 for y (landscape displays). Applying these factors to prescaled data results in numbers from approximately 0 to 799 (for x) and 0 to 1023 (for y) for portrait displays, and vice versa for landscape displays. (Note that the prescaled data allows a 5% overlap, as mentioned in the preceding subsection.)

The default scale factors provide for touchpad data corresponding to the whole screen. Smaller scale factors reduce the area to which the touchpad maps, thereby allowing you to more finely tune the cursor position. This also applies to mouse movement, allowing changes in the apparent motion sensitivity of the device.

Setting the Origin

The origin is the point denoted by the upper left corner of the touchpad, in absolute and absolute/relative mode. In relative mode, the origin has no meaning. By default, the touchpad origin corresponds to the upper left corner of the screen, that is, the point 0,0 in raster units. By changing the origin, you can use the touchpad (in absolute mode) to correspond to a portion of the screen.

This feature is useful for applications that need to move the cursor within a fixed window, rather than across the whole screen. For example, a program that displays a menu

in one window might set the origin to the upper left corner of the menu window. Consequently, the touchpad maps onto the menu window instead of the entire screen.

Hysteresis

The hysteresis value defines the dimensions of a box around your finger position on the touchpad or the current position of the mouse. Movement within the box does not change the position of the cursor on the screen.

Specify the hysteresis value in raster units. The touchpad manager compares the value to the difference between the current and previous finger positions on the touchpad or the current and previous positions of the mouse. If the difference is less than the hysteresis value, the cursor does not move. If the difference is greater than the hysteresis value, the hysteresis value is subtracted from the difference and the cursor moves the resulting distance. The default hysteresis value is five.

EXAMPLES

Display current characteristics.

```
$ tpm
  Mode: absolute
  Xscale: 1024, Yscale: 800
  Hysteresis: 5
  Origin: 0, 0
```

Set characteristics to absolute/relative mode with half the default scaling sensitivity (portrait display).

```
$ tpm -ar -s 400 512
```

NAME

tr_font – transliterate characters within a font

SYNOPSIS

tr_font *font_name hex_conversion_table*

DESCRIPTION

The **tr_font** command allows you to change the order in which characters appear in fonts. It rearranges the graphic images associated with the characters in *font_name*, according to information in the *hex_conversion_table*. Use it if you create a new font file from two font files that have different character orders.

tr_font only works on fonts already formatted for SR10. It works directly on the font, without creating a backup.

The format for the *hex_conversion_table* file is:

```
src_ordinal dest_ordinal comment
src_ordinal dest_ordinal comment
src_ordinal dest_ordinal comment
```

where *src_ordinal* is the hexadecimal ordinal value of the character whose graphic image is to be used as the source, *dest_ordinal* is the ordinal value of the character which gets the transliterated image, and *comment* is an optional remark (for the ASCII character set, the hexadecimal ordinal value 41 represents the character 'A'). If the font was created by concatenating two fonts with **cvt_font**, then the hexadecimal ordinal value of the lowest possible character in the second font is 80.

EXAMPLE

The following example rearranges the characters in the SR10 format font file named **courier** according to the information in the *hex_conversion_table* **theirs_to_ours**.

```
$ tr_font courier theirs_to_ours
```

This is a sample of a *hex_conversion_table* file.

```
A1 A1 !down
A2 A2 cent
A3 A3 sterling
A5 A5 yen
A7 A7 section
A8 A4 currency
AB AB guillemotleft
B6 B6 paragraph
```

TR_FONT

Aegis

TR_FONT

B7 B7 bullet
B8 B8 quitesinglebase
BB BB guill

SEE ALSO

More information is available. Type
help cvt_font

NAME

ts – display the module name and time stamp

SYNOPSIS

ts [-nhd] *object_module_name*

DESCRIPTION

ts displays the time stamp and module name stored in an object module. Shown is the time and date that the module was created by one of the linkers or compilers. The time stamp is not affected by copying an object module, so it is a reliable indicator of whether particular object modules are copies of one another.

Note: There is a homonymous DM command: **ts** – Set tab stops for all windows. Type **help ts_dm** for details about that command.

OPTIONS

-nhd Option does not print the table header. **ts** outputs in tabular format with table header by default.

NAME

tz – set or display system time zone

SYNOPSIS

tz [*tz_name* | *utc_delta* [*new_tz*]]

DESCRIPTION

tz sets the system time zone to a known time zone or to an offset from Coordinate Universal Time (utc).

To set the actual time registered by the nodes's internal clock, use the **calendar** command. If no arguments are specified, **tz** displays the current setting.

tz_name (optional) Specify new time zone. The following are valid names:

Name	Time Zone
EDT	Eastern Daylight Time
EST	Eastern Standard Time
CDT	Central Daylight Time
CST	Central Standard Time
MDT	Mountain Daylight Time
MST	Mountain Standard Time
PDT	Pacific Daylight Time
PST	Pacific Standard Time
GMT	Greenwich Mean Time
UTC	Coordinated Universal Time

Default if omitted: use *utc_delta* argument

utc_delta (optional) Specify positive or negative offset from utc. The plus sign is optional for positive offsets. Format for offset is *hh:mm* (for example, -10:00 for ten hours earlier than, west of, Coordinated Universal Time). Only whole or half hour offsets may be specified. Other fractional offsets produce an error message.

Default if omitted: use *tz_name* argument

new_tz (optional) Specify new time zone name to be assigned to the zone indicated by the *utc_delta* argument. Use this argument to create time zones that are not included in the list above.

Default if omitted: no name assigned

EXAMPLES

Display current time zone.

```
$ tz
  Timezone:      EST
  Delta from UTC: -5:00
```

Set time zone to Pacific Daylight Time.

```
$ tz pdt
```

Create (and set) a time zone named GST that is four and a half hours later than (east of) Coordinated Universal Time.

```
$ tz 4:30 gst
```

SEE ALSO

More information is available. Type
help calendar

NAME

uctnode – uncatalog a node

SYNOPSIS

/etc/uctnode [options] pathname ...

DESCRIPTION

uctnode removes the specified entry directory name from the local copy of the network root directory. After the name is removed, objects cataloged under that node's entry directory are no longer accessible to you or other nodes on the network.

If you use the **-root** option, the nodename is also removed from the network's replicated root directory.

Node entry directories are created with the **ctnode** command.

pathname (required) Specify node entry directory name to be uncataloged. Multiple pathnames and wildcarding are permitted.

OPTIONS

- l** List directory names as they are uncataloged.
- root** Uncatalog the node in the network root as well as in the the local root directory.

EXAMPLES

Uncatalog the node with the entry directory name specified.

```
$ /etc/uctnode als_node
```

SEE ALSO

More information is available. Type

help ctnode For details about cataloging nodes

NAME

uctob -uncatalog the specified pathname, without deleting the associated object.

SYNOPSIS

/etc/uctob [-br] pathname...

DESCRIPTION

The command **uctob** removes the specified pathname from the name space. The object associated with the pathname is not affected. This command is primarily intended for system-level debugging use.

OPTIONS

-br Suppress listing of names and uids of objects as they are uncataloged. These are reported unless this option is specified.

EXAMPLES

```
$ /etc/uctob testfile
"testfile" uid is 16791C0C.40000074.
$
```

This example uncatalogs **testfile**.

SEE ALSO

More information is available. Type

help ctob For details about cataloging nodes

NAME

uk_to_iso – convert files to ISO format

SYNOPSIS

uk_to_iso *input_file output_file*

DESCRIPTION

These utilities convert files written with the overloaded 7-bit national fonts to the International Standards Organization (ISO) 8-bit format. The overloaded fonts include any with a specific language suffix (for example, *f7x13.french*, or *din_f7x11.german*). If you created and/or edited a file using one of the national fonts, that file is a candidate for conversion.

You are not required to convert files, but probably will want to because

1. The overloaded fonts replace certain ASCII characters with national ones, have that subset of ASCII characters and the national characters in one file. The 8-bit fonts available as of SR10 include all the ASCII characters and the national characters.
2. The 8-bit fonts also include a wider range of national characters, so you can enter any character in any western European language. This was not always possible with the overloaded fonts. For example, there was not enough space in the overloaded font to include all the French characters, but they all exist in the 8-bit fonts.

There are two parameters to the conversion utilities. You must provide the name of the file you want to convert (*input_file*) and your *output_file*. If *output_file* already exists, the utilities abort.

The default location for the utilities is */usr/apollo/bin*.

FILES

<i>/usr/apollo/bin/french_to_iso</i>	Converts overloaded French to ISO format
<i>/usr/apollo/bin/german_to_iso</i>	Converts overloaded German to ISO format
<i>/usr/apollo/bin/nor.dan_to_iso</i>	Converts overloaded Norwegian/Danish to ISO format
<i>/usr/apollo/bin/swedish_to_iso</i>	Converts overloaded Swedish/Finnish to ISO format
<i>/usr/apollo/bin/swiss_to_iso</i>	Converts overloaded Swiss to ISO format
<i>/usr/apollo/bin/uk_to_iso</i>	Converts overloaded U.K. English to ISO format

DIAGNOSTICS

All messages are generally self-explanatory.

NAME

ulkob – unlock an object

SYNOPSIS

/etc/ulkob [options] [pathname ...]

DESCRIPTION

ulkob unlocks objects residing on, or locked by processes running on, the current node. You cannot unlock objects on remote nodes unless you locked them (see **-f** below).

This command can be used when a program terminates abnormally, leaving objects locked, or to unlock objects previously locked with the **lkob** command.

To obtain a list of your node's locked objects, use the **llkob** command.

pathname (optional)

Specify name of object to be unlocked. Multiple pathnames and wild-carding are permitted.

Default if omitted: **-u** option must be specified

OPTIONS

If no options are specified, the object is unlocked for all lock modes.

- r** Unlock an object that was locked for read mode; the lock must be owned by this process.
- w** Unlock an object that was locked for write mode; the lock must be owned by this process.
- i** Unlock an object that was locked for reading with intent to write; the lock must be owned by this process.
- f** Forcibly unlock an object. It may have been locked for any mode and the lock may be owned by any process. The object must reside on the current node, however, or must have been locked by the current node. In other words, you cannot unlock objects on a remote node unless you locked them.
- l** List the name of each object as it is unlocked.
- u uid ...** Specify the UID of the object(s) to unlock. Multiple UIDs are permitted. If the *pathname* argument is omitted, then this option is required.

EXAMPLES

Forcibly unlock the file **mary** for any mode and unlock the two objects with the specified UIDs.

```
$ /etc/ulkob mary -f
```

```
$ /etc/ulkob -uid 1C1A9E2F.20000246 1C1A9E42.50000246
```

SEE ALSO

More information is available. Type

help lkob For details about locking objects

help llkob For details about listing those objects that are locked

NAME

umask – set UNIX file-creation-mode mask

SYNOPSIS

umask [*nnn*]

DESCRIPTION

umask sets or displays your UNIX file-creation-mode mask. This is an internal shell command.

ARGUMENTS

nnn (optional) Specify the read/write/execute permissions for owner, group, and others, respectively. The value of each specified octal digit is subtracted from the corresponding "digit" specified by the system for the creation of a file.

Default if omitted: display current mask value

EXAMPLES

To remove write permission of the group and others, execute the following command.

```
$ umask 022
```

Files normally created with mode 777 become mode 755; files created with mode 666 become mode 644.

NAME

uuid_gen – UUID generating program

SYNOPSIS

`/etc/ncs/uuid_gen [-c] [-p] [-C] [-P]`

DESCRIPTION

The `uuid_gen` program generates Universal Unique Identifiers (UUIDs). Without options, it generates a character-string representation of a UUID. The options enable you to generate templates for Network Interface Definition Language (NIDL) files or to generate source-code representations of UUIDs, suitable for initializing variables of type `uuid_t`.

OPTIONS

- `-c` Generate a template, including a UUID attribute, for an interface definition in the C syntax of NIDL.
- `-p` Generate a template, including a UUID attribute, for an interface definition in the Pascal syntax of NIDL.
- `-C` Generate a C source-code representation of a UUID.
- `-P` Generate a Pascal source-code representation of a UUID.

EXAMPLES

Generate a template for an interface definition in the C syntax of NIDL:

```
$ /sys/ncs/uuid_gen -c
%c
[
uuid(34dc239ec000.0d.00.00.7c.5f.00.00.00),
version(1)
]
interface INTERFACENAME {
```

Generate a C source-code representation of a UUID:

```
$ /sys/ncs/uuid_gen -C
= { 0x34dc23af,
0xf000,
0x0000,
0x0d,
{0x00, 0x00, 0x7c, 0x5f, 0x00, 0x00, 0x00} };
```

NAME

vctl – set/display VT100 terminal characteristics

SYNOPSIS

vctl [options]

DESCRIPTION

vctl allows you to set or display information about how the VT100 terminal emulator driver handles input from the keyboard (for example, whether it echoes characters, or how it interprets key sequences typed at the keyboard).

This command is valid only if you have the VT100 terminal emulation software package running on your node. In addition, vctl can be run only in a window where the VT100 emulator is already running.

OPTIONS

If no options are specified, the current VT100 settings are displayed.

- default Set all options to their default values. This allows you to quickly reset values to known states.

- [no]cvt_in_nl Convert a newline (linefeed) to a carriage return on input. The initial setting is –nocvt_in_line.

- [no]cvt_in_cr Convert a carriage return to a newline on input. The initial setting is –cvt_in_cr.

- [no]cvt_out_nl Convert a newline to carriage return, newline on output. The initial setting is –cvt_out_nl.

- [no]cvt_out_cr Convert a carriage return to a newline on output. The initial setting is –nocvt_out_cr.

- [no]echo Turn the echoing of input characters on or off. The initial setting is echo.

- [no]echo_ctl Turn the echoing of control characters (such as CTRL/Z) on or off. The initial setting is noecho_ctl.

- [no]echo_erase If echo is on, controls whether characters are visibly erased from the screen when the erase character is typed. The combination of echo and noecho_erase causes the erase character to be echoed until all characters on a line are erased. The initial setting is –echo_erase.

- [no]raw** If raw mode is on, a program reading from the keyboard in the VT100 receives each character as it is typed. If "raw" mode is off, such a program blocks until a full line has been typed. A full line is a sequence of characters ending in a newline character. In other words, in "non-raw" mode, a program blocks until a carriage return or line feed is typed.
- [no]echo_kill** If echo is on, this option controls whether a line is visibly erased from the screen when the line kill-character is typed. The combination of echo and noecho_kill causes the kill character to be echoed and a new line to be displayed. The initial setting is `-echo_kill`.
- eof *char*** Set the end-of-file character. The EOF character is initially set to CTRL/Z.
- erase *char*** Set the erase character. This option is valid only when data is being passed to the terminal emulator in "cooked" mode. The *char* can be any character or one-byte hexadecimal value. You may have to enclose some characters in quotation marks in the shell. The erase character is initially set to backspace (08 hex).
- intr *char*** Set the interrupt character, which sends an interrupt fault to the process group of the terminal emulator. The interrupt character is initially set to CTRL/C.
- kill *char*** Set the kill character. This option is valid only when data is being passed to the emulator in cooked mode. The kill character is initially set to CTRL/X.
- quit *char*** Set the quit character. The quit character is initially set to CTRL/Q.
- suspend *char*** Set the suspend character. The suspend character is initially set to hex FF, which is equivalent to its being disabled.
- [no]enable_sigs** If `enable_sigs` is on, the fault-generating characters (interrupt, quit, suspend) have their special meaning. If `enable_sigs` is off, then these characters are not treated specially.
- eol *char*** Set the extra break character. The EOL character is initially set to hex FF, which is equivalent to its being disabled. If it is enabled, the EOL character behaves like CR in that any program reading from the keyboard will immediately wake up and read whatever has been typed so far, including the EOL character itself.

EXAMPLES

Display current settings.

```
$ vctl
Erase (character delete) character: ""H" (08 hex)
Kill (line delete) character: ""U" (15 hex)
End of file character: ""Z" (1A hex)
Interrupt character: ""C" (03 hex)
Quit character: ""Q" (11 hex)
Extra break character: FF (hex)
Suspend character: FF (hex)
Raw: FALSE,   Echo: TRUE,   Echo_Erase: TRUE
Echo_Kill: TRUE,   Echo_Ctl: FALSE, Cvt_In_CR: TRUE
Cvt_In_NL: FALSE, Cvt_Out_NL: TRUE, Cvt_Out_CR: FALSE
Enable_Sigs: TRUE
$
```

Set quit character to hex FE, enable conversion of output newlines to carriage returns.

```
$ vctl --quit 0FE --cvt_out_cr
```

SEE ALSO

More information is available. Type

help vt100 for information on the VT100 terminal emulator.

NAME

voff – deactivate the shell's **-v** flag

SYNOPSIS

voff

DESCRIPTION

voff turns off the shell's **-v** (verify) flag, which is turned on by the **von** command or the **-v** option on the **sh** command line. When the flag is off, command lines are not displayed when they are read by the shell. Verification is off by default.

voff requires no arguments or options.

SEE ALSO

More information is available. Type

help von	For details about turning shell input verification on
help sh	For details about the shell command-line interpreter
help shell	For general shell information

NAME

von – activate the shell's **-v** flag

SYNOPSIS

von

DESCRIPTION

von turns on input verification. As commands are executed, or comments processed, they are written to the error output stream of the shell. In shell scripts, you can use **von** to show the progress being made by the script.

If **von** is turned on in a shell script, it remains on until that shell script exits, or until overridden by a **voff** in a nested shell script. When a shell script exits, the state of input verification is returned to the state in effect just before the script was invoked.

von requires no arguments or options.

SEE ALSO

More information is available. Type

help voff	For details about turning shell input verification off
help sh	For details about the shell command line interpreter
help shell	For general information about the shell

NAME

vsize – set/display VT100 window settings

SYNOPSIS

vsize [*options*]

DESCRIPTION

The **vsize** command allows you to set the dimensions of the VT100 emulator window pane. This command is valid only from within the VT100 emulator (which is invoked with the VT100 command); attempting to use it directly from the shell causes an error.

OPTIONS

If no options are specified, **vsize** displays the current window pane settings.

- l n** Specify the height of the window pane in lines. If this option is omitted, the height remains unchanged.
- c n** Specify the width of the window in columns. If this option is omitted, the width remains unchanged.
- std** Set the height of the window to 24 lines and the width to 80 columns. This is the same as saying **-l 24 -c 80**.

EXAMPLES

Invoke VT100 emulator and Display current settings.

```
$ vt100
$ vsize
Screen size is 18 lines by 70 columns.
```

Change the width. Exit the emulator and return to the shell.

```
$ vsize -c 60
Old screen size is 18 lines by 70 columns.
New screen size is 18 lines by 60 columns.
$ *** EOF ***
$
```

NAME

vt100 – VT100 terminal emulator

SYNOPSIS

vt100 [*options*] [*pathname* [*arg1 arg2 ...*]]

DESCRIPTION

The **vt100** command creates a window running the VT100 terminal emulator and starts up a shell within the window.

The VT100 terminal emulation package is intended for use with two types of programs. When used in conjunction with remote communications packages such as Domain TCP/IP or X.25, the VT100 terminal emulator allows you to interact with the remote system as if you were logged into a VT100 connected to that system. Using the VT100 terminal emulator with programs that take advantage of VT100 special features allows you to run these programs on a Domain node without having to tailor them to the Domain environment.

If any options are specified, they must precede the argument(s).

pathname [*arg1 arg2 ...*] (optional)

Specify the name of a command or program for the shell in the VT100 window to invoke. You must give the full pathname; for example, */com/ld*. *arg1, arg2, ...* are valid arguments to the selected command (or program): for example, */com/ld //my_node/my_home_dir*. The default is to invoke the value of the variable *\$SHELL*, or if *\$SHELL* is not set, invoke */com/sh*.

OPTIONS

If any options are specified, they must precede the argument(s). Once **vt100** is running, you may change the window size with the **vszie** command.

- std** Set up a VT100 window that is 24 lines by 80 columns (the standard size of a VT100 screen).
- lines *n*** Set up a VT100 window with the number of lines specified by *n*. The number of lines cannot exceed the number of lines in the DM window running the VT100 emulator.
- columns *n*** Set up a VT100 window with the number of columns specified by *n*. The number of columns cannot exceed the number of columns of the DM window running the VT100 emulator.

The VT100 terminal emulation package consists of the following:

- The terminal emulation software, which performs the functions of a VT100 terminal, such as handling VT100-type escape sequences. The terminal emulator redirects the handling of keyboard input and screen output to stream manager operations. The terminal emulator is invoked within a DM window by the **vt100** shell command.

- The terminal emulator driver, which performs keyboard input functions such as erasing or echoing characters. The `vt1` shell command allows you to set and display the VT100 terminal characteristics controlled by the terminal emulator driver.

EXAMPLES

1. Create a window running the VT100 emulator and start a shell running within the window.

```
$ vt100
```

2. Open a connection to the remote system specified by *hostname* and create a window running the VT100 emulator.

```
$ vt100 login hostname
```

KEYBOARD LAYOUT

The table below shows how the keys on a Domain low-profile keyboard map to the keys of a VT100. This assumes that you are running the VT100 Keyboard Emulation package on your node. Note that the VT100 definitions for the F2, F3, and F7 keys supersede the usual `emt` definitions for these keys.

Domain key	Vt100 keypad
<INS MODE>	<ESC>
<CHAR DEL>	<RUBOUT>
<F2>	<PF1>
<F3>	<PF2>
<F4>	<PF3>
<F5>	<PF4>
SHIFT/<F2>	<7>
SHIFT/<F3>	<8>
SHIFT/<F4>	<9>
SHIFT/<F5>	<->
CTRL/<F2>	<4>
CTRL/<F3>	<5>
CTRL/<F4>	<6>
CTRL/<F5>	<->
<F6>	<1>
<F7>	<2>
SHIFT/<F6>	<3>
SHIFT/<F7>	<ENTER>
CTRL/<F6>	<0>
CTRL/<F7>	<->

SEE ALSO

More information is available. Type

- | | |
|------------------------|--|
| help vt100 unix | For information about using VT100 with a remote UNIX system running |
| help vctl | For information about setting VT100 terminal emulator characteristics |
| help vsize | For information about changing the dimensions of the VT100 emulator window |

NAME

wbak – create a magnetic media backup file

SYNOPSIS

```
wbak -f fileno [-dev | m[unit] | f | ct]
      [-full|-incr|-af dtm|-bef dtm]
      [-fid id] [-own id] [-vid vol_id]
      [-sla|-nsla] [-wla|-nwla] [-nhi] [-pdtu]
      [-reo] [-reten|-nreten] [-no_eot]
      [-sysboot] [-l|-ld|-lf|-ll] [-to filename]
      [-type uasc|unstruct|hdru]
      [-r] [-stdout] [-presr10] pathname...
```

DESCRIPTION

wbak writes one or more objects to either a removable media, disk file or standard output. These objects may be directory trees, files, or links. For each object, the information saved includes the name, object data, and attributes associated with the object, such as the access control list. This lets you reconstruct files, the directory tree, or any portion of the tree using the **rbak** command.

The **wbak** and **rbak** commands are intended both for disk backup and for interchanging information between separate Apollo installations. Use the **rwmt** command to read and write magnetic media that are used for interchanging information with non-Apollo installations.

pathname (required) Specify the name of the object to be written to backup media. This may be a directory, file, or link. If it is a file, then the file is written as specified. If it is a link, then the link is resolved and the resolution object is written to backup media. If it is a directory, all subordinate files and subdirectories in the tree are written. Note that the *pathname* specified reflects the way the directory is stored on the backup media, and that the same name must be used when reading files using *pathnames* in **rbak**. Multiple *pathnames* and wildcarding are permitted. If you omit this argument, **wbak** will prompt you for it. You may specify a hyphen (-) as an argument to direct **wbak** to standard input for further arguments and options.

OPTIONS

The **-f** option is required, as it specifies where on the backup media the new file is to be written. If you omit it, **wbak** will prompt you for it.

Tape File Identifiers

-fid *file_id* Specify a 1-17 character file ID to be written in the file header label for use when writing a file to a labeled volume. If this option is omitted, the file is not named and can only be restored by the file number.

-f [*position*] Specify the file position for the write operation. Valid values for *position* are *cur*, *end*, or a nonzero integer. A position of *cur* specifies that the file should be written at the current position on the backup media; the media must have been previously written by *wbak* and its position must not have been disturbed.

A position of *end* specifies that the file should be written at the end of the backup media file set. This causes *wbak* to append the specified disk file (*pathname* argument) to the very end of the file set.

A position specified by a nonzero integer value causes the file to be written at that absolute position in the backup media volume. If multiple *pathname* arguments are supplied, the value of *position* is incremented by one after each file has been written.

The default value for *position* is 1.

Mode Control

The object specified by the *pathname* argument must be a directory for either **-full** or **-incr** to have meaning.

-full (default) Specify a full backup; save all files in specified trees.

-incr Specify an incremental backup; save files that were modified since the last backup recorded in the *backup_history* file stored in the *pathname* directory.

-af *dtm* Save all files modified after the given date and time; *dtm* is in the form *yy/mm/dd.hh:mm*. The date defaults to today, and the time to midnight if either of those are omitted from *dtm*.

-bef *dtm* Save all files last modified before the given date and time.

Label Control

-wla (default) Write the backup media volume label if the backup file number is 1.

-nwla Suppress writing of the backup media volume label.

-own *id* Specify backup media volume owner (1-14 character name). This option is only meaningful when used with the **-wla** option.

-vid *vol_id* Specify a 1-6 character volume ID for use when labeling a volume. This option is only meaningful when the backup file number is 1. The default volume ID is '' (blank).

- sla (default)** Display the label information written for this backup file on standard output.
- nsla** Suppress output of label information.

Listing Control

You may include the **-l** option, or any combination of **-ld**, **-lf**, and **-ll**.

- l** Write the names of all files, directories, and links saved to standard output.
- lf** Write the names of all files saved to standard output.
- ld** Write the names of all directories saved to standard output.
- ll** Write the names of all links saved to standard output.

Backup Device Control

- dev *d[unit]*** Specify device type and unit number. *d* must be either **m** (for reel-to-reel magnetic tape), **ct** (for cartridge tape), or **f** (for floppy), depending on which drive is being used. *unit* is an integer (0-3). Both are required for reel-to-reel tapes (that is, **-dev m2**). A unit number is not required for floppy disks and cartridge tapes (that is, **-dev f**). If this option is omitted, **rbak** assumes device **m0**.

CAUTION: Floppy disk support for this command is limited. In particular, error detection during reads and writes is poor. do not use this command with floppy disks when the data being placed on the floppy disks are critical and unrecoverable.

- to *filename*** Backup output is written to the specified streams object rather than removable media. This can then be restored using the **-from** option in **rbak**. If the file already exists, use the **-r** option to replace it. If **-type** option is not specified the file will be assigned the default type. You cannot use the **-file *n*** option with streams.

- type [*uasc* | *unstruct* | *hdru*]** Specify the type of the object *filename*. It can be one of ASCII (**uasc**), Unstructured (**unstruct**) or Streams header-undefined (**hdru**) type.

- r** If the object specified with the **-to** option already exists, this option allows it to be replaced. The type of *filename* is however left unchanged.

- stdout** The backup output is written to standard output.
- reo** Force previous backup media volume to be reopened, and suppress reading of backup media volume label. Use only when backup media has not been repositioned since last **wbak** or **rbak**.

Special Cartridge Tape Control Options

- reten** Retension the cartridge tape (unwind to the end, then rewind). This can be helpful if you have encountered cartridge tape reading errors. Retensioning requires about 1.5 minutes to complete.
- nreten** (default) Do not retension the cartridge tape.
- no_eot** Suppress the writing of two tape marks at the end of the tape file, which are the standard signal for end of tape. The cartridge can't position between the two tapemarks to be ready for a successive call to **wbak** (as it does on magtape), without rewinding the tape and searching forward, so this option speeds up multiple invocations of **wbak**. It should not be used on the last invocation of **wbak**. Also, **-f cur** should be used on all **wbak** invocations in a series except the first one.
- sysboot** Permit use of a bootable tape that has a special boot program at the beginning. This option causes **wbak** to skip over the first file on the tape. This option is only necessary when the first file on the tape is being written (**-f l**).

Miscellaneous Control Options

- nhi** Suppress update of the backup history file.
- (hyphen)** Read standard input for further arguments or options; input is accepted until **wbak** receives an EOF.
- pdtu** Preserves the last date/time-used information on objects. After each object is backed up on tape, the date/time-used information is reset to the value it had before the backup.
- presr10** Allows you to make a tape with pre-SR10 format from an SR10 node. This tape will have no ACLs by default. You can restore it to a pre-SR10 volume by means of the pre-SR10 **rbak**. If you make a tape without this option it will not be readable on a pre-SR10 system.

EXAMPLES

```
$ wbak //mask/wby -f 1 -af 87/11/19.12.00 -fid wby -L
```

This command writes the directory //mask/wby to tape. The directory is written out to tape file one, and the file ID wby is written to the file's label. Disk files from directory wby are written to the tape only if they have been modified since noon on November 19, 1987. The label and the names of the files written to tape are printed to standard output.

When this command is executed, wbak writes the following information to standard output:

```
Label:
  File number:    1
  File section:  1
  File id:        wby
  Date written:   1987/11/20 10:47:58 EST
```

```
Starting write:
```

```
(file) "//mask/wby/among" written
(file) "//mask/wby/school" written
(file) "//mask/wby/children" written
(file) "//mask/wby/backup_history" written
(dir)  "//mask/wby/" written.
```

```
Write complete.
```

This command backs up the entire contents of the node whose entry directory name is goeey. Note that the file ID is specified as "node 27 backup" to make it easy to identify when you want to reload it, and that the command assigns volume and owner IDs.

```
$ wbak -f 1 -own "john doe" -vid "volbk2" -fid "node 27 backup" //goeey
```

When this command is executed, `wbak` writes the following information to standard output:

```
Label:
  Volume id:      VOLBK2
  Owner id:       john doe
  File number:    1
  File section:   1
  File id:        n 27 backup
  File written:   1987/02/17 18:00:39 EST
```

Starting write:

Write complete.

This command uses wildcards to match only those files in the `ug` subdirectory of the current working directory whose names begin with the letters `a` through `f` and end with `_example`.

```
$ wbak -f 1 -own "john doe" -vid "volbk1" ug/[a-f?]*_example -l
```

When this command is executed, `wbak` writes the following information to standard output:

```
Label:
  Volume id:      VOLBK1
  Owner id:       john doe
  File number:    1
  File section:   1
  File id:        (no id specified)
  File written:   1988/02/17 17:58:52 EST
```

Starting write:

```
(file) "ug/cmf_example" written.
(file) "ug/cmt_example" written.
(file) "ug/cpboot_example" written.
(file) "ug/cpf_example" written.
(file) "ug/cpt_example" written.
(file) "ug/fpat_example" written.
(file) "ug/fppmask_example" written.
(file) "ug/fst_example" written.
```

Write complete.

```
$ wbak src -to /backup/bck_out.file
```

This command writes the backup output for the directory `src` to the file `/fred/bck_out.file`. The directory can be restored in either of the following two ways :

```
rbak src -from /backup/bck_out.file
or
catf /fred/bck_out.file | rbak src -stdin
```

Using streams as a backup output media, it is possible to stage the backup output to intermediate disks and then use `rwmt` to write the intermediate file to the magnetic tape. The sequence to use is as follows

```
$ wbak //otter -to //backup/ot wbak //otter -to //backup/tmp1
```

This writes the backup output to an intermediate file `//backup/tmp1` followed by

```
rwmt -f 2 -w //backup/tmp1 -raw -rl 8192 -nobs -ansi
```

When the magtape unit is available at a later time the intermediate file is written to the magtape. Note that it is **ESSENTIAL** to use the `-raw`, `-rl 8192` and the `-nobs` options of `rwmt`, for `rbak` to be able to read the backup from tape. All tapes used for this must have the **ANSI** specified volume label. You can only use this sequence for magnetic tapes. `rbak` will not be able to restore data written using the above sequence for cartridge tapes instead of magnetic tapes. This sequence has exactly the same effect as using

```
wbak //otter -dev mt -f 2
```

You can then use `rbak` as follows to retrieve the data

```
rbak //otter -f 2 -dev mt
```

SEE ALSO

More information is available. Type

help rbak	For information on restoring or indexing a magnetic media backup file
help rwmt	For information on reading/writing foreign magtapes
help media	For information on removable media

NAME

wd – set or display the current working directory

SYNOPSIS

wd [*pathname*]

DESCRIPTION

wd sets the working directory for the current process to the specified directory. The working directory is where the system looks for objects when you don't explicitly specify a directory as a part of a pathname.

ARGUMENTS

pathname (optional)

Specify new working directory. This may be a derived name, but must point to a directory or link to a directory. Specifying a file causes an error. **wd** also accepts the command-line parser arguments - and *.

Default if omitted: display current working directory

EXAMPLES

Set new working directory. Display the new setting.

```
$ wd //fred/release_4
$ wd
//fred/release_4
```

Set working directory with derived name. Display the new setting.

```
$ wd stuff/revised
$ wd
//fred/personal/stuff/revised
```

Direct input to a file named `newdir` that holds the name of the new working directory.

```
$ wd *newdir
$
```

SEE ALSO

More information is available. Type

help pathname For general information about pathnames

help cl For information about the command-line parser

NAME

while – execute a while loop

SYNOPSIS

```
while condition
do
    command ...
enddo
```

DESCRIPTION

while executes a command (or commands) as long as the results of a Boolean test are true. You can extend the **while** command over several lines if you use it interactively or in a shell script. When you use **while** interactively, and extend the command over more than one line, the shell prompts you for each new line of the command with the **\$_** prompt.

ARGUMENTS

condition (required)

Specify a command or program to execute and test for truth, or specify a variable expression or Boolean variable to test for truth. "Truth" usually means that the command executes successfully (without any errors), or that a shell variable expression or Boolean is "true". (Specifically, this argument is evaluated true if it returns an abort-severity level of 0 (zero).)

Refer to *Using Your Aegis Environment* for more information on shell variables.

command ... (required)

Specify the command(s) or program(s) to execute if *condition* returns true.

EXAMPLES

```
$ eon
$ K := 3
$ while (('K > 0))
$ _do args (('k'); k := `k - 1
$ _enddo
3
2
1
$
```

WHILE

Aegis

WHILE

SEE ALSO

More information is available. Type

help abtsev For more information on abort-severity levels

help exit For details about exiting from a while loop

NAME

xdmc – execute a DM command from the shell

SYNOPSIS

xdmc *dm_command* [*args...*]

DESCRIPTION

xdmc allows you to invoke Display Manager commands from the command shell or from within a shell script. This is similar to pressing the CMD key on the keyboard and then typing the DM command in the DM input window, which is the usual way to perform DM operations.

dm_command (required)

Specify the Display Manager command to be executed. Type **help dm commands** for a topical command index.

args ... (optional)

Specify any arguments to be passed to the DM command. These are sent directly to the DM without further processing by the command shell.

Default if omitted: no arguments passed

EXAMPLES

\$ xdmc dq

Cause the DM to send a quit fault to the current process.

This is analagous to the **sigp** (**signal_process**) shell command, with the following important difference. Whereas **sigp** accepts an argument designating which process to fault, this example sends a fault to any process that is pointed to by the current cursor position.

\$ xdmc cp /com/sh

Cause the DM to create a new process and invoke the shell. This is the same as pressing **<SHELL>**.

SEE ALSO

More information is available. Type

help dm For general information about the Display Manager

help dm commands For a topical index of DM commands

NAME

xoff – deactivate the shell's **-x** flag

SYNOPSIS

xoff

DESCRIPTION

xoff turns off the shell's **-x** (execution trace) flag, which is turned on by the **xon** command or by the **-x** option on the **sh** command. When the flag is off, command lines are not displayed before execution. The flag is off by default.

xoff requires no arguments or options.

SEE ALSO

More information is available. Type

- help xon** For details about turning execution tracing on
- help sh** For details about the shell command-line interpreter
- help shell** For general shell information

NAME

xon – activate the shell's **-x** flag

SYNOPSIS

xon

DESCRIPTION

xon turns on execution tracing. Just before each command is executed, its full pathname and arguments are written to the error output stream of the shell. In shell scripts, **xon** can be used to show the progress being made by the script, and can help debug shell scripts by showing the actual arguments being passed to commands, after all shell processing on them is complete.

By default, execution tracing is off when a shell is invoked.

If **xon** is turned on in a shell script, it remains on until that shell script exits, or until over-ridden by an **xoff** in a nested shell script. When a shell script exits, the state of execution tracing is returned to the state in effect just before the script was invoked.

xon requires no arguments or options.

SEE ALSO

More information is available. Type

help xoff For details about turning execution tracing off

help sh For details about the shell command line interpreter

help shell For general shell information

NAME

xsubs – run shell-script subsystem manager

SYNOPSIS

xsubs *pathname* [*args...*]

DESCRIPTION

Once a protected subsystem, a subsystem manager(s), and a subsystem data object(s) exist, any user can execute the manager program. To run a binary manager program, simply execute the program. To run a shell-script manager program, you must use the **xsubs** command. Note that in order to see the name of a subsystem created on another node, you must copy the file `/sys/subsys/subsystem_name` to your node. If you do not copy this file, you can use the subsystem managers to operate on the objects, but when you ask to display the name of the subsystem, you get an error message like the following:

```
$ subs //fred/jtj/com/top_secret
?(subs) Can't show subsystem manager type
      for "//fred/jtj/com/top_secret"
- subsystem name not found (US/aclm)
$
```

ARGUMENTS

pathname (required)

Specify shell script containing the subsystem manager to be executed. Note that this script must contain the commands **subs -up** and **subs -down** in order to enter and exit the subsystem.

args ... (optional)

Specify arguments to be passed to the shell script.

Default if omitted: no arguments passed

EXAMPLES

Suppose you have an append-only list that you wish to protect. Anyone can read the list, and append to the list, but no one can overwrite existing contents. Assume that the subsystem **append_only** already exists. Then the **app** shell script, which appends standard input to an append-only file, looks like this:

```
# app --- append to an append_only file
subs -up
catf >>^1      # append to the file passed as first argument
subs -down
```

To make **app** a manager of the **append_only** subsystem, enter

```
$ensubs append_only          # enter subsystem
subs app append_only -mgr
*** EOF ****
```

A run of **app** looks like this:

```
$xsubs app aofile            # execute app on aofile
this is the stuff that is appended
*** EOF ***
```

SEE ALSO

More information is available. Type

help protected_subsystems

For a list of shell commands for use with protected subsystems

help protection protected_subsystems

For a detailed description of protected subsystems

Reader's Response

Please take a few minutes to send us the information we need to revise and improve our manuals from your point of view.

Document Title: *Aegis Command Reference*

Order No.: 002547-A00

Date of Publication: July, 1988

What type of user are you?

- | | |
|--|---|
| <input type="checkbox"/> System programmer; language _____ | |
| <input type="checkbox"/> Applications programmer; language _____ | |
| <input type="checkbox"/> System maintenance person | <input type="checkbox"/> Manager/Professional |
| <input type="checkbox"/> System Administrator | <input type="checkbox"/> Technical Professional |
| <input type="checkbox"/> Student Programmer | <input type="checkbox"/> Novice |
| <input type="checkbox"/> Other | |

How often do you use the Apollo system? _____

What parts of the manual are especially useful for the job you are doing? _____

What additional information would you like the manual to include? _____

Please list any errors, omissions, or problem areas in the manual. (Identify errors by page, section, figure, or table number wherever possible. Specify additional index entries.) _____

Your Name Date

Organization

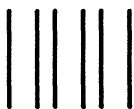
Street Address

City State Zip

No postage necessary if mailed in the U.S.

cut or fold along dotted line

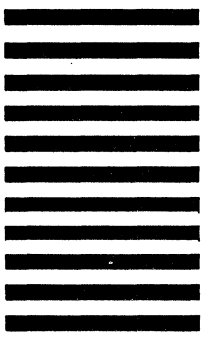
fold



NO POSTAGE
NECESSARY
IF MAILED
IN THE
UNITED STATES

BUSINESS REPLY MAIL
FIRST CLASS PERMIT NO. 78 CHELMSFORD, MA 01824
POSTAGE WILL BE PAID BY ADDRESSEE

APOLLO COMPUTER INC.
Technical Publications
P.O. Box 451
Chelmsford, MA 01824



fold

Reader's Response

Please take a few minutes to send us the information we need to revise and improve our manuals from your point of view.

Document Title: *Aegis Command Reference*

Order No.: 002547-A00

Date of Publication: July, 1988

What type of user are you?

- | | |
|--|---|
| <input type="checkbox"/> System programmer; language _____ | |
| <input type="checkbox"/> Applications programmer; language _____ | |
| <input type="checkbox"/> System maintenance person | <input type="checkbox"/> Manager/Professional |
| <input type="checkbox"/> System Administrator | <input type="checkbox"/> Technical Professional |
| <input type="checkbox"/> Student Programmer | <input type="checkbox"/> Novice |
| <input type="checkbox"/> Other | |

How often do you use the Apollo system? _____

What parts of the manual are especially useful for the job you are doing? _____

What additional information would you like the manual to include? _____

Please list any errors, omissions, or problem areas in the manual. (Identify errors by page, section, figure, or table number wherever possible. Specify additional index entries.) _____

Your Name _____ Date _____

Organization _____

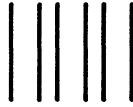
Street Address _____

City _____ State _____ Zip _____

No postage necessary if mailed in the U.S.

cut or fold along dotted line

fold

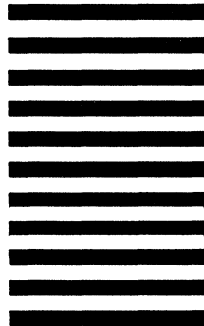


NO POSTAGE
NECESSARY
IF MAILED
IN THE
UNITED STATES

BUSINESS REPLY MAIL
FIRST CLASS PERMIT NO. 78 CHELMSFORD, MA 01824

POSTAGE WILL BE PAID BY ADDRESSEE

APOLLO COMPUTER INC.
Technical Publications
P.O. Box 451
Chelmsford, MA 01824



fold

Reader's Response

Please take a few minutes to send us the information we need to revise and improve our manuals from your point of view.

Document Title: *Aegis Command Reference*

Order No.: 002547-A00

Date of Publication: July, 1988

What type of user are you?

- | | |
|--|---|
| <input type="checkbox"/> System programmer; language _____ | |
| <input type="checkbox"/> Applications programmer; language _____ | |
| <input type="checkbox"/> System maintenance person | <input type="checkbox"/> Manager/Professional |
| <input type="checkbox"/> System Administrator | <input type="checkbox"/> Technical Professional |
| <input type="checkbox"/> Student Programmer | <input type="checkbox"/> Novice |
| <input type="checkbox"/> Other | |

How often do you use the Apollo system? _____

What parts of the manual are especially useful for the job you are doing? _____

What additional information would you like the manual to include? _____

Please list any errors, omissions, or problem areas in the manual. (Identify errors by page, section, figure, or table number wherever possible. Specify additional index entries.) _____

Your Name Date

Organization

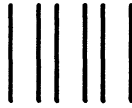
Street Address

City State Zip

No postage necessary if mailed in the U.S.

cut or fold along dotted line

fold



NO POSTAGE
NECESSARY
IF MAILED
IN THE
UNITED STATES

BUSINESS REPLY MAIL
FIRST CLASS PERMIT NO. 78 CHELMSFORD, MA 01824

POSTAGE WILL BE PAID BY ADDRESSEE

APOLLO COMPUTER INC.
Technical Publications
P.O. Box 451
Chelmsford, MA 01824



fold



002547-A00