```
**********************************************************
   63646 Wed Jun  6 19:50:09 2012
new/usr/src/cmd/format/menu_command.c
2741 format shouldn't allow write SMI label to disk with EFI partition
**********************************************************
______unchanged_portion_omitted_

1459 /*
1460  * This routine implements the 'label' command.  It writes the
1461  * primary and backup labels onto the current disk.
1462  */
1463 int
1464 c_label()
1465 {
1466         int                     status;
1467         int                     deflt, *defltptr = NULL;

1469         /*
1470          * There must be a current disk type (and therefore a current disk).
1471          */
1472         if (cur_dtype == NULL) {
1473                 err_print("Current Disk Type is not set.\n");
1474                 return (-1);
1475         }
1476         /*
1477          * The current disk must be formatted to label it.
1478          */
1479         if (!(cur_flags & DISK_FORMATTED)) {
1480                 err_print("Current Disk is unformatted.\n");
1481                 return (-1);
1482         }
1483         /*
1484          * Check for a valid fdisk table entry for Solaris
1485          */
1486         if (!good_fdisk()) {
1487                 return (-1);
1488         }
1489         /*
1490          * Check to see if there are any mounted file systems anywhere
1491          * on the current disk.  If so, refuse to label the disk, but
1492          * only if the partitions would change for the mounted partitions.
1493          *
1494          */
1495         if (checkmount((diskaddr_t)-1, (diskaddr_t)-1)) {
1496                 /* Bleagh, too descriptive */
1497                 if (check_label_with_mount()) {
1498                         err_print("Cannot label disk while it has "
1499                             "mounted partitions.\n\n");
1500                         return (-1);
1501                 }
1502         }

1504         /*
1505          * check to see if there any partitions being used for swapping
1506          * on the current disk.  If so, refuse to label the disk, but
1507          * only if the partitions would change for the mounted partitions.
1508          */
1509         if (checkswap((diskaddr_t)-1, (diskaddr_t)-1)) {
1510                 if (check_label_with_swap()) {
1511                         err_print("Cannot label disk while its "
1512                             "partitions are currently being used for "
1513                             "swapping.\n");
1514                         return (-1);
1515                 }
1516         }
```



```
1518         /*
1519          * Check to see if any partitions used for svm, vxvm or live upgrade
1520          * are on the disk. If so, refuse to label the disk, but only
1521          * if we are trying to shrink a partition in use.
1522          */
1523         if (checkdevinuse(cur_disk->disk_name, (diskaddr_t)-1,
1524             (diskaddr_t)-1, 0, 1)) {
1525                 err_print("Cannot label disk when "
1526                     "partitions are in use as described.\n");
1527                 return (-1);
1528         }

1530         /*
1531          * If there is not a current partition map, warn the user we
1532          * are going to use the default.  The default is the first
1533          * partition map we encountered in the data file.  If there is
1534          * no default we give up.
1535          */
1536         if (cur_parts == NULL) {
1537                 fmt_print("Current Partition Table is not set, "
1538                     "using default.\n");
1539                 cur_disk->disk_parts = cur_parts = cur_dtype->dtype_plist;
1540                 if (cur_parts == NULL) {
1541                         err_print("No default available, cannot label.\n");
1542                         return (-1);
1543                 }
1544         }
1545         /*
1546          * If expert (-e) mode, then ask user if they wish
1547          * to change the current solaris label into an EFI one
1548          */
1549         if (expert_mode) {
1550 #if defined(_SUNOS_VTOC_8)
1551                 int             i;
1552 #endif
1553                 int             choice;
1554                 u_ioparam_t             ioparam;
1555                 struct extvtoc  vtoc;
1556                 struct dk_label label;
1557                 struct dk_gpt   *vtoc64;
1558                 struct efi_info efinfo;
1559                 struct disk_type        *dptr;

1561                 /* Ask user what label to use */
1562                 fmt_print("[0] SMI Label\n");
1563                 fmt_print("[1] EFI Label\n");
1564                 ioparam.io_bounds.lower = 0;
1565                 ioparam.io_bounds.upper = 1;
1566                 if ((cur_label == L_TYPE_SOLARIS) &&
1567                     (cur_disk->fdisk_part.systid != EFI_PMBR))
1568                         deflt = L_TYPE_SOLARIS;
1566                 if (cur_label == L_TYPE_SOLARIS)
1567                         deflt = 0;
1569                 else
1570                         deflt = L_TYPE_EFI;
1569                         deflt = 1;
1571                 defltptr = &deflt;
1572                 choice = input(FIO_INT, "Specify Label type", ':',
1573                     &ioparam, defltptr, DATA_INPUT);
1574                 if ((choice == L_TYPE_SOLARIS) &&
1575                     (cur_label == L_TYPE_SOLARIS) &&
1576                     (cur_disk->fdisk_part.systid != EFI_PMBR)) {
1573                 if ((choice == 0) && (cur_label == L_TYPE_SOLARIS)) {
1577                         goto expert_end;
1578                 } else if ((choice == L_TYPE_EFI) &&
1579                     (cur_label == L_TYPE_EFI)) {
```

```
1575                } else if ((choice == 1) && (cur_label == L_TYPE_EFI)) {
1580                        goto expert_end;
1581                }
1582                switch (choice) {
1583                case L_TYPE_SOLARIS:
1579                case 0:
1584                /*
1585                 * EFI label to SMI label
1586                 */
1587                if (cur_dtype->capacity > INFINITY) {
1588                        fmt_print("Warning: SMI labels only support up to "
1589                            "2 TB.\n");
1590                }

1592                if (cur_disk->fdisk_part.systid == EFI_PMBR) {
1593                        fmt_print("Warning: This disk has an EFI label. "
1594                            "Changing to SMI label will erase all\n"
1595                            "current partitions.\n");
1596                        if (check("Continue"))
1597                        return (-1);
1598 #if defined(_FIRMWARE_NEEDS_FDISK)
1599                        fmt_print("You must use fdisk to delete the current "
1600                            "EFI partition and create a new\n"
1601                            "Solaris partition before you can convert the "
1602                            "label.\n");
1603                        return (-1);
1604 #endif
1605                }

1607 #if defined(_FIRMWARE_NEEDS_FDISK)
1608                if (!(((cur_disk->fdisk_part.systid != SUNIXOS) ||
1609                    (cur_disk->fdisk_part.systid != SUNIXOS2)) &&
1610                    (cur_disk->fdisk_part.numsect > 0))) {
1611                        fmt_print("You must use fdisk to create a Solaris "
1612                            "partition before you can convert the label.\n");
1613                        return (-1);
1614                }
1615 #endif

1617                (void) memset((char *)&label, 0, sizeof (struct dk_label));

1619                (void) strcpy(x86_devname, cur_disk->disk_name);
1620                if (cur_ctype->ctype_ctype == DKC_DIRECT)
1621                        dptr = auto_direct_get_geom_label(cur_file,  &label);
1622                else
1623                        dptr = auto_sense(cur_file, 1, &label);
1624                if (dptr == NULL) {
1625                        fmt_print("Autoconfiguration failed.\n");
1626                        return (-1);
1627                }

1629                pcyl = label.dkl_pcyl;
1630                ncyl = label.dkl_ncyl;
1631                acyl = label.dkl_acyl;
1632                nhead = label.dkl_nhead;
1633                nsect = label.dkl_nsect;

1635                if (delete_disk_type(cur_disk->disk_type) == 0) {
1636                        cur_label = L_TYPE_SOLARIS;
1637                        cur_disk->label_type = L_TYPE_SOLARIS;
1638                        cur_disk->disk_type = dptr;
1639                        cur_disk->disk_parts = dptr->dtype_plist;
1640                        cur_dtype = dptr;
1641                        cur_parts = dptr->dtype_plist;

1643                        if (status = write_label())
```



```
1644                                err_print("Label failed.\n");
1645                        else
1646                                cur_disk->disk_flags &= ~DSK_LABEL_DIRTY;

1648                        return (status);
1649                } else {
1650                        err_print("Label failed.\n");
1651                        return (-1);
1652                }


1655                case L_TYPE_EFI:
1651                case 1:
1656                /*
1657                 * SMI label to EFI label
1658                 */

1660                if ((cur_disk->fdisk_part.systid == SUNIXOS) ||
1661                    (cur_disk->fdisk_part.systid == SUNIXOS2)) {
1662                        fmt_print("Warning: This disk has an SMI label. "
1663                            "Changing to EFI label will erase all\ncurrent "
1664                            "partitions.\n");

1657                fmt_print("Warning: This disk has an SMI label. Changing to "
1658                    "EFI label will erase all\ncurrent partitions.\n");

1665                        if (check("Continue")) {
1666                                return (-1);
1667                        }
1668                }

1670                if (get_disk_info(cur_file, &efinfo) != 0) {
1671                        return (-1);
1672                }
1673                (void) memset((char *)&label, 0, sizeof (struct dk_label));
1674                label.dkl_pcyl = pcyl;
1675                label.dkl_ncyl = ncyl;
1676                label.dkl_acyl = acyl;
1677 #if defined(_SUNOS_VTOC_16)
1678                label.dkl_bcyl = bcyl;
1679 #endif                 /* defined(_SUNOC_VTOC_16) */
1680                label.dkl_nhead = nhead;
1681                label.dkl_nsect = nsect;
1682 #if defined(_SUNOS_VTOC_8)
1683                for (i = 0; i < NDKMAP; i++) {
1684                        label.dkl_map[i] = cur_parts->pinfo_map[i];
1685                }
1686 #endif                 /* defined(_SUNOS_VTOC_8) */
1687                label.dkl_magic = DKL_MAGIC;
1688                label.dkl_vtoc = cur_parts->vtoc;
1689                if (label_to_vtoc(&vtoc, &label) == -1) {
1690                        return (-1);
1691                }
1692                if (SMI_vtoc_to_EFI(cur_file, &vtoc64) == -1) {
1693                        return (-1);
1694                }
1695                if (efi_write(cur_file, vtoc64) != 0) {
1696                        err_check(vtoc64);
1697                        err_print("Warning: error writing EFI.\n");
1698                        return (-1);
1699                } else {
1700                        cur_disk->disk_flags &= ~DSK_LABEL_DIRTY;
1701                }
1702                /*
1703                 * copy over the EFI vtoc onto the SMI vtoc and return
1704                 * okay.
```

```
1705                  */
1706                 dptr = auto_efi_sense(cur_file, &efinfo);
1707                 if (dptr == NULL) {
1708                         fmt_print("Autoconfiguration failed.\n");
1709                         return (-1);
1710                 }

1712                 cur_label = L_TYPE_EFI;
1713                 cur_disk->label_type = L_TYPE_EFI;
1714                 cur_disk->disk_type = dptr;
1715                 cur_disk->disk_parts = dptr->dtype_plist;
1716                 cur_dtype = dptr;
1717                 cur_parts = dptr->dtype_plist;
1718                 cur_parts->etoc = vtoc64;

1720                 ncyl = pcyl = nsect = psect = acyl = phead = 0;

1722                 /*
1723                  * Get the Solais Fdisk Partition information.
1724                  */
1725                 (void) copy_solaris_part(&cur_disk->fdisk_part);

1727                 return (0);
1728                 }
1729         }

1731 expert_end:
1732         /*
1733          * Make sure the user is serious.
1734          */
1735         if (check("Ready to label disk, continue")) {
1736                 return (-1);
1737         }
1738         /*
1739          * Write the labels out (this will also notify unix) and
1740          * return status.
1741          */
1742         fmt_print("\n");
1743         if (status = write_label())
1744                 err_print("Label failed.\n");
1745         return (status);
1746 }
```

_____***unchanged_portion_omitted_***