Appendix A - The sys_config script, On Line Configuration Backups

The /usr/local/bin/sys_config script and an associated cron entry are configured on each system at installation. Typically the sys_config script is run just after midnight each day. The output of this script is saved to the local machine in /usr/local/bin/system_configs. This output is also copied to administration systems in two buildings, this functionality is contained in a separate script.

```
!/bin/ksh
#
#
HOSTID=`hostname`
# This variable was added for gathering printer information.
SAVEDIR=/usr/local/bin/system_configs/$HOSTID/var/sam/lp
#
PATH=/etc:/bin:/usr/bin/$PATH
DATE=`date '+%y%m%d'`
#
OUTFILE=/usr/local/bin/system_configs/`uname -a | awk '{ print $2 }'`_config.$DATE
#
#
# Lets gather some general system info and save it to $OUTFILE
#
/usr/bin/echo $DATE > $OUTFILE
/usr/bin/echo " " >> $OUTFILE
/usr/bin/echo "The script that created this is /usr/local/bin/sys_config" >> $OUTFILE
/usr/bin/echo " " >> $OUTFILE
#
# The following commands and files are in no particular order. If you identify
# additional commands or files the would be useful simply add them in double
# quotes.

for i in "ioscan -fun" "uname -a" "lvlnboot -v" "lanscan" "swapinfo" "bdf" "cat /etc/fstab" "cat
/etc/nsswitch.conf" "crontab -l" "cat /var/adm/cron/cron.allow" "cat /var/adm/cron/cron.deny" "cat
/var/adm/cron/at.allow" "cat /var/adm/cron/at.deny" "/usr/sbin/swlist -l product" "cat /etc/passwd" "cat
/etc/group" "cat /etc/exports" "cat /etc/ntp.conf" "ls -al /dev/*/group" "cat /etc/sudoers"  "cat /etc/hosts" "cat
/etc/services" "cat /etc/profile" "cat /etc/inetd.conf" "cat /etc/oratab" "cat /etc/listener.ora" "cat /etc/sybtab"
"cat /etc/sybdbas" "cat /etc/inittab" "cat /overlord/.rhosts" "cat /etc/resolv.conf" "ll /sbin/rc0.d" "ll
/sbin/rc1.d" "ll /sbin/rc2.d" "ll /sbin/rc3.d" "ll /sbin/rc4.d" "cat /etc/ddfa/dp" "cat /etc/ddfa/pcf" "cat
/etc/ftpusers" "cat /etc/ftpusers.exclude" "/usr/sbin/setboot" "cat /etc/hosts.allow" "cat /etc/hosts.deny" "cat
/etc/syslog.conf" "cat /etc/lvmrc" "/usr/sbin/lsdev" "cat /stand/system"

do

/usr/bin/echo "Output of "$i >> $OUTFILE

$i >> $OUTFILE

/usr/bin/echo " " >> $OUTFILE
/usr/bin/echo " " >> $OUTFILE

done
#
# now create a system file from the running kernel just in case the one you
```

```
# did a cat of above is not correct

echo " " >> $OUTFILE

echo "Output of /usr/lbin/sysadm/system_prep -p /tmp/system" >> $OUTFILE

echo " " >> $OUTFILE

/usr/lbin/sysadm/system_prep -s /tmp/system

cat /tmp/system >> $OUTFILE

echo " " >> $OUTFILE
#
# now lets gather lifls and a lifcp (AUTO) information for all 3 boot disks
# in the system

      /usr/sbin/lvlnboot -v | grep "Boot Disk" | awk '{ print $1 }' | awk '{ FS="/"; print "/"$2"/r"$3"/"$4 }' |
      while read boot_device ;
      do
      /usr/bin/echo "" >> $OUTFILE
      /usr/bin/echo "Lifls on "$boot_device >> $OUTFILE
          /usr/bin/echo "" >> $OUTFILE
      /usr/bin/lifls $boot_device >> $OUTFILE
          /usr/bin/echo "" >> $OUTFILE
          /usr/bin/echo "Lifcp on AUTO for "$boot_device >> $OUTFILE
          /usr/bin/echo "" >> $OUTFILE
          /usr/bin/lifcp $boot_device:AUTO - >> $OUTFILE
          /usr/bin/echo "" >> $OUTFILE
      done

# now lets do cats of all executables in /sbin/init.d

/usr/bin/echo " " >> $OUTFILE

echo "CONFIG INFO IN /sbin/init.d" >> $OUTFILE

for i in `ls /sbin/init.d`
do
echo "======================" >> $OUTFILE
echo $i >> $OUTFILE
cat /sbin/init.d/$i >> $OUTFILE
done
#
#
#now lets gather all the config info in /etc/rc.config.d
echo "CONFIG INFO IN /etc/rc.config.d" >> $OUTFILE
for i in `ls /etc/rc.config.d`
do
echo "======================" >> $OUTFILE
echo $i >> $OUTFILE
cat /etc/rc.config.d/$i >> $OUTFILE
done
#
#
```

```
echo "The following sections are the output of vgdisplay -v and" >> $OUTFILE
echo "lvdisplay -v commands for each volume group on the system." >> $OUTFILE
#
ls /dev/*/group | awk '{FS="/";print "/dev/"$3, $3 }' |
while read fullpath vgname;
        do
        for i in $fullpath
        do
        vgcfgbackup $fullpath
        vgexport -p -m /etc/lvmconf.map/map.$vgname -v $vgname
        vgdisplay -v $fullpath >> $OUTFILE
        ls -al $fullpath/* >> $OUTFILE
        vgdisplay -v $fullpath | grep "LV Name" | awk '{ print $3 }' |
        while read lvpath;
        do
          lvdisplay -v $lvpath >> $OUTFILE
          echo " " >> $OUTFILE
          echo " " >> $OUTFILE
        done
        echo " " >> $OUTFILE
        echo " " >> $OUTFILE
        done
done
#
#
# This section was added to gather printer configurations
#
/usr/sam/lbin/lpmgr -S -xsavedir=${SAVEDIR}
#
cp /etc/ioconfig /usr/local/bin/system_configs/$HOSTID/etc/ioconfig.$DATE
cp /etc/lvmtab /usr/local/bin/system_configs/$HOSTID/etc/lvmtab.$DATE
cp /stand/ioconfig /usr/local/bin/system_configs/$HOSTID/stand/ioconfig.$DATE
cp /stand/vmunix /usr/local/bin/system_configs/$HOSTID/stand/vmunix.$DATE
#
cd /etc/lvmconf
for i in `ls *.conf`
do
cp /etc/lvmconf/$i /usr/local/bin/system_configs/$HOSTID/vgcfg/$i.$DATE
done
for i in `ls /etc/lvmconf.map`
do
cp /etc/lvmconf.map/$i /usr/local/bin/system_configs/$HOSTID/vgexport/$i.$DATE
done
```

Appendix B - lv_mergeit and lv_splitit

The lv_mergeit and lv_splitit scripts are put in place at system installation. The syntax or order of the lvsplit and lvmerge commands are significant. These scripts are used to manage the tertiary boot disk.

/usr/local/bin/lv_mergeit

```
#!/bin/ksh
lvmerge /dev/vg00/lvol1b /dev/vg00/lvol1
lvmerge /dev/vg00/lvol2b /dev/vg00/lvol2
lvmerge /dev/vg00/lvol3b /dev/vg00/lvol3
lvmerge /dev/vg00/lvol4b /dev/vg00/lvol4
lvmerge /dev/vg00/lvol5b /dev/vg00/lvol5
lvmerge /dev/vg00/lvol6b /dev/vg00/lvol6
lvmerge /dev/vg00/lvol7b /dev/vg00/lvol7
```

/usr/local/bin/lv_splitit

```
#!/bin/ksh
lvsplit /dev/vg00/lvol7 /dev/vg00/lvol6 /dev/vg00/lvol5 /dev/vg00/lvol4 /dev/vg00/lvol2 /dev/vg00/lvol1
/dev/vg00/lvol3
```

Appendix C – Secure_it RC configuration

The /etc/rc.config.d/secure_it and /sbin/init.d/secure_it scripts are put into place on each system at installation. These scripts follow the standard rc configuration with one exception, there is no “K” link to execute at system shutdown. Much of the functionality of this script has been removed for presentation in this document.

/etc/rc.config.d/secure_it

```
#!/bin/ksh
#
# This script determines if the /sbin/init.d/secure_it script runs at boot
# time.
#
# secure_it contains administrative tools to maintain security standards
# across reboots.
#
# There is no kill functionality in this script, therefore there is not a
# "K" link for this script.
#
SECURE_IT=1
```

/sbin/init.d/secure_it

```
#!/bin/ksh
#
#
# NOTE:  This script should be the last script run in the RC set.
#
```

```
#     It will only run at system boot time not at shutdown.
## ------------------------------------------------------------------------
#
# Allowed exit values:
#       0 = success; causes "OK" to show up in checklist.
#       1 = failure; causes "FAIL" to show up in checklist.
#       2 = skip; causes "N/A" to show up in the checklist.
#           Use this value if execution of this script is overridden
#           by the use of a control variable, or if this script is not
#           appropriate to execute for some other reason.

# Input and output:
#       stdin is redirected from /dev/null
#
#       stdout and stderr are redirected to the /etc/rc.log file
#       during checklist mode, or to the console in raw mode.

PATH=/usr/sbin:/usr/bin:/sbin
export PATH

# NOTE: If your script executes in run state 0 or state 1, then /usr might
#       not be available.  Do not attempt to access commands or files in
#       /usr unless your script executes in run state 2 or greater.  Other
#       file systems typically not mounted until run state 2 include /var
#       and /opt.

rval=0

# Check the exit value of a command run by this script.  If non-zero, the
# exit code is echoed to the log file and the return value of this script
# is set to indicate failure.

set_return() {
        x=$?
        if [ $x -ne 0 ]; then
                echo "EXIT CODE: $x"
                rval=1  # script FAILed
        fi
}

case $1 in
'start_msg')
        echo "Configuration and Security Script is running."
        ;;


'start')
        # source the variables
        if [ -f /etc/rc.config.d/secure_it ] ; then
                . /etc/rc.config.d/secure_it
        else
                echo "ERROR: /etc/rc.config.d/secure_it file MISSING"
        fi

        # Check to see if this script is allowed to run...
```

```
        if [ "$SECURE_IT" != 1 ]; then
              rval=2
        else

           if [ -d /usr/local/bin ] ; then
             /usr/bin/echo "Maintaining permissions on /usr/local/bin, OK"
             /usr/bin/chmod 755 /usr/local/bin
             /usr/bin/chown bin:bin /usr/local/bin
             else
             /usr/bin/echo "/usr/local/bin/does not exist, CHECK IT"
           fi
           if [ -f /var/adm/syslog/syslog.log ] ; then
             /usr/bin/chmod 640 /var/adm/syslog/syslog.log
             /usr/bin/chown root:root /var/adm/syslog/syslog.log
             /usr/bin/chmod 640 /var/adm/syslog/OLDsyslog.log
             /usr/bin/chown root:root /var/adm/syslog/OLDsyslog.log
             /usr/bin/echo "Maintaining perm-ownership of /var/adm/syslog.log, OK"
             else
             /usr/bin/touch /var/adm/syslog/syslog.log
             /usr/bin/chmod 640 /var/adm/syslog/syslog.log
             /usr/bin/chown root:root /var/adm/syslog/syslog.log
             /usr/bin/chmod 640 /var/adm/syslog/OLDsyslog.log
             /usr/bin/chown root:root /var/adm/syslog/OLDsyslog.log
             /usr/bin/echo "/var/adm/syslog.log did not exist creating it, CHECK IT"
        fi

           if [ -f /usr/contrib/bin/nettune ] ; then
           /usr/contrib/bin/nettune -s tcp_random_seq 2
           /usr/bin/echo "nettune is setting tcp_random_seq, OK"
           else
           /usr/bin/echo "/usr/contrib/bin/nettune not found, CHECK IT"
        fi

        if [ -f /etc/motd ] ; then
           /usr/bin/chmod 644 /etc/motd
           /usr/bin/chown root:sys /etc/motd
           /usr/bin/echo "maintaining perm-ownership of /etc/motd, OK"
           else
           /usr/bin/echo "/etc/motd does not exist, CHECK IT"
        fi

if [ -f /var/adm/btmp ] ; then
           /usr/bin/chmod 600 /var/adm/btmp
           /usr/bin/chown adm:adm /var/adm/btmp
           /usr/bin/echo "maintaining perm-ownership of /var/adm/btmp, OK"
           else
           /usr/bin/echo "/var/adm/btmp does not exist creating it, CHECK IT:"
           /usr/bin/touch /var/adm/btmp
           /usr/bin/chmod 600 /var/adm/btmp
           /usr/bin/chown adm:adm /var/adm/btmp
        fi

        if [ -f /var/adm/wtmp ] ; then
           /usr/bin/chmod 644 /var/adm/wtmp
           /usr/bin/chown adm:adm /var/adm/wtmp
```

```
        /usr/bin/echo "maintaining perm-ownership of /var/adm/wtmp, OK"
        else
        /usr/bin/echo "/var/adm/wtmp does not exist creating it, CHECK IT:"
        /usr/bin/touch /var/adm/wtmp
        /usr/bin/chmod 644 /var/adm/wtmp
        /usr/bin/chown adm:adm /var/adm/wtmp
    fi

    if [ -f /etc/utmp ] ; then
        /usr/bin/chmod 644 /etc/utmp
        /usr/bin/chown adm:adm /etc/utmp
        /usr/bin/echo "maintaining perm-ownership of /etc/utmp, OK"
        else
        /usr/bin/echo "/etc/utmp does not exist creating it, CHECK IT:"
        /usr/bin/touch /etc/utmp
        /usr/bin/chmod 644 /etc/utmp
        /usr/bin/chown adm:adm /etc/utmp
    fi


      fi
   ;;

esac

exit $rval
```

Appendix D – Monitor_setuid Script

/usr/local/bin/monitor_setuid

```
#!/bin/ksh
#
# Created by Daniel hartline
# This script will monitor changes, deletions, creation of setuid, setgid and
# sticky files and directories
# on each system. After a new installation of HP-UX or patching an existing system this
# script should be run with the create_standard option. This will create a baseline of all
# setuid and setgid files/dirs on the system. The script will then be run from cron each
# night with the compare_standard option. This option compares current setuid and
# setgid files/dirs against the current standard for the system. If differences are found
# emails are sent to the administration team.
#
# the code for defining exit values exists in this script but at the time of creation
# we are not utilizing it.
#

# Allowed exit values:
#       0 = success
#       1 = failure
#       2 = skip; causes "N/A" to show up in the checklist.
#       Use this value if execution of this script is overridden
#          by the use of a control variable, or if this script is not
#          appropriate to execute for some other reason.
```

```
PATH=/usr/sbin:/usr/bin:/sbin
export PATH

rval=0

# Check the exit value of a command run by this script.  If non-zero, the
# exit code is echoed to the log file and the return value of this script
# is set to indicate failure.

set_return() {
        x=$?
        if [ $x -ne 0 ]; then
          echo "EXIT CODE: $x"
          rval=1    # script FAILed
        fi
}


case $1 in

'create_standard')

# Create new setuid/setgid standard file, do not run actual diff to check
# for changes/problems

DATE=`date '+%c'`
SETUIDPATH=/usr/local/bin/security
SETUID_OUT=${SETUIDPATH}/SETUID.OUT
SETUID_GREP=${SETUIDPATH}/SETUID.GREP
SETUID_STANDARD=${SETUIDPATH}/.SETUID.STANDARD
SETUID_STANDARD_HIST=${SETUIDPATH}/.SETUID.STANDARD.HIST

if [ -f ${SETUID_STANDARD} ] ;
then
/usr/bin/rm ${SETUID_STANDARD}
/usr/bin/find / \( -perm -4000 -o -perm -2000 -o -perm -1000 \) -fstype vxfs -exec /usr/bin/ll -ald {} \; | awk
'{ print $1, $3, $4, $9 }' > ${SETUID_OUT}
/usr/bin/grep -v "/var/opt/dce/rpc/local" ${SETUID_OUT} > ${SETUID_GREP}
/usr/bin/sort -o ${SETUID_STANDARD} ${SETUID_GREP}
/usr/bin/rm ${SETUID_OUT}
/usr/bin/rm ${SETUID_GREP}
/usr/bin/chmod 600 ${SETUID_STANDARD}
else
/usr/bin/find / \( -perm -4000 -o -perm -2000 -o -perm -1000 \) -fstype vxfs -exec /usr/bin/ll -ald {} \; | awk
'{ print $1, $3, $4, $9 }' > ${SETUID_OUT}
/usr/bin/grep -v "/var/opt/dce/rpc/local" ${SETUID_OUT} > ${SETUID_GREP}
/usr/bin/sort -o ${SETUID_STANDARD} ${SETUID_GREP}
/usr/bin/rm ${SETUID_OUT}
/usr/bin/rm ${SETUID_GREP}
/usr/bin/chmod 600 ${SETUID_STANDARD}
fi

/usr/bin/echo "New SETUID Standard Created on $DATE" >> ${SETUID_STANDARD_HIST}
```

```
# DO A MORE OF THE STANDARD FILE HERE SO EACH TIME STANDARDS ARE
# created they con be verified before they are actually used as standards

/usr/bin/echo ""
/usr/bin/echo ""
/usr/bin/echo "The standard file you just created will now be displayed via the"
/usr/bin/echo "more command. Verify that it does not contain unusual setuid, setgid or sticky"
/usr/bin/echo "files or directories. If unusual files or directories are observed communicate this to"
/usr/bin/echo "the lead."
/usr/bin/echo ""
/usr/bin/echo "Sleeping for 8 seconds"
/usr/bin/echo ""
/usr/bin/sleep 8

if [ -s ${SETUID_STANDARD} ] ;
then
/usr/bin/echo "============  VERIFY SETUID, SETGID AND STICKY FILES AND DIRECTORIES
================"
/usr/bin/echo ""
/usr/bin/more ${SETUID_STANDARD}
/usr/bin/echo ""
/usr/bin/echo ""
else
/usr/bin/echo "SETUID Standard was not successfully created, CHECK IT."
fi

'compare_standard')

# verify existance of standard file, if it does not exist create it and email.
# Compare current setuid/setgid files and dirs to current standard. If differences are found
# email admin team.

HOSTID=`hostname`
DATE=`date '+%y%m%d'`
SETUIDPATH=/usr/local/bin/security
SETUID_STANDARD=${SETUIDPATH}/.SETUID.STANDARD
SETUID_OUT=${SETUIDPATH}/SETUID.OUT
SETUID_OUT_C=${SETUIDPATH}/SETUID.OUT_C
SETUID_GREP=${SETUIDPATH}/SETUID.GREP
SETUID_COMPARE=${SETUIDPATH}/setuid.compare.${DATE}
SETUID_DIFF=${SETUIDPATH}/setuid.diff.${DATE}

# If the standard file does not exist create it and email, nothing to compare.

if [ -s ${SETUID_STANDARD} ] ;
then

# find current setuid and setgid and compare to standard.

        /usr/bin/find / \( -perm -4000 -o -perm -2000 -o -perm -1000 \) -fstype vxfs -exec /usr/bin/ll -ald {}
\; | awk '{ print $1, $3, $4, $9 }' > ${SETUID_OUT_C}
        /usr/bin/grep -v "/var/opt/dce/rpc/local" ${SETUID_OUT_C} > ${SETUID_GREP}
        /usr/bin/sort -o ${SETUID_COMPARE} ${SETUID_GREP}
        /usr/bin/rm ${SETUID_OUT_C}
        /usr/bin/rm ${SETUID_GREP}
```

```
        /usr/bin/chmod 600 ${SETUID_COMPARE}
        /usr/bin/diff ${SETUID_STANDARD} ${SETUID_COMPARE} > ${SETUID_DIFF}
        /usr/bin/chmod 600 ${SETUID_DIFF}

        if [ -s ${SETUID_DIFF} ] ;
        then
        echo "###############" >> ${SETUID_DIFF}
        echo "First file passed to diff command is standard file, <" >> ${SETUID_DIFF}
        echo "Second file passed to diff command is compare file, >" >> ${SETUID_DIFF}
        echo "###############" >> ${SETUID_DIFF}
        echo "Created by /usr/local/bin/monitor_setuid" >>  ${SETUID_DIFF}
        /usr/bin/mailx -s "SECURITY: setuid output from $HOSTID" INSERT EMAIL ADDRESSES
HERE < ${SETUID_DIFF}
        else
        echo " "
        fi

else

/usr/bin/find / \( -perm -4000 -o -perm -2000 -o -perm -1000 \) -fstype vxfs -exec /usr/bin/ll -ald {} \; | awk
'{ print $1, $3, $4, $9 }' > ${SETUID_OUT}
/usr/bin/grep -v "/var/opt/dce/rpc/local" ${SETUID_OUT} > ${SETUID_GREP}
/usr/bin/sort -o ${SETUID_STANDARD} ${SETUID_GREP}
/usr/bin/rm ${SETUID_OUT}
/usr/bin/rm ${SETUID_GREP}
/usr/bin/chmod 600 ${SETUID_STANDARD}

SETUID_EMAIL=${SETUIDPATH}/setuid.email
/usr/bin/echo "The SETUID.STANDARD file did not exist and a " >  ${SETUID_EMAIL}
/usr/bin/echo "compare_standard was attempted. This email was " >> ${SETUID_EMAIL}
/usr/bin/echo "created by the /usr/local/bin/monitor_setuid " >> ${SETUID_EMAIL}
/usr/bin/echo "script. " >> ${SETUID_EMAIL}

/usr/bin/mailx -s "Setuid Standard file did not exist on $HOSTID created it" INSERT EMAIL
ADDRESSES HERE < ${SETUID_EMAIL}

/usr/bin/rm ${SETUID_EMAIL}
fi

        ;;

*)
        echo "usage: $0 {create_standard|compare_standard}"
        rval=1
        ;;
esac

exit $rval
```

Appendix E – Monitor_config script

```
#!/bin/ksh
#
# Created by Daniel hartline
# This script will monitor a number of configuration aspects that should not change over
# time. This monitoring will consist of a standard being created and used to compare
# current output of commands. If a discrepency is identified the administration team
# will be notified via email.
# If a configuration is modified the standard should be recreated by running this script
# with the create_standards option.
#
# the code for defining exit values exists in this script but at the time of creation
# we are not utilizing it.
#

# Allowed exit values:
#         0 = success
#         1 = failure
#         2 = skip; causes "N/A" to show up in the checklist.
#         Use this value if execution of this script is overridden
#            by the use of a control variable, or if this script is not
#            appropriate to execute for some other reason.

PATH=/usr/sbin:/usr/bin:/sbin
export PATH

rval=0

# Check the exit value of a command run by this script.  If non-zero, the
# exit code is echoed to the log file and the return value of this script
# is set to indicate failure.

set_return() {
        x=$?
        if [ $x -ne 0 ]; then
          echo "EXIT CODE: $x"
          rval=1    # script FAILed
        fi
}


case $1 in

##############################################################################################
###

'create_standard')

# SET VARIABLES ##############
DATE=`date '+%c'`
```

```
CONFIGPATH=/usr/local/bin/security
LVLNBOOT_STANDARD=${CONFIGPATH}/.LVLNBOOT.STANDARD
ALL_STANDARD_HIST=${CONFIGPATH}/.ALL.STANDARD.HIST

# REMOVE EXISTING STANDARD FILES IF THEY EXIST AND CREATE NEW ONES
######################

if [ -f ${LVLNBOOT_STANDARD} ] ;
then
/usr/bin/rm ${LVLNBOOT_STANDARD}
/usr/sbin/lvlnboot -v | /usr/bin/grep -v alternate > ${LVLNBOOT_STANDARD}
/usr/bin/chmod 600 ${LVLNBOOT_STANDARD}
else
/usr/sbin/lvlnboot -v | /usr/bin/grep -v alternate > ${LVLNBOOT_STANDARD}
/usr/bin/chmod 600 ${LVLNBOOT_STANDARD}
fi
/usr/bin/echo "New CONFIG/ALL Standards Created on $DATE" >> ${ALL_STANDARD_HIST}

# DO A MORE OF THE STANDARD FILES HERE SO EACH TIME STANDARDS ARE
# created they con be verified before they are actually used as standards

/usr/bin/echo ""
/usr/bin/echo ""
/usr/bin/echo "The standard files you just created will now be displayed via the"
/usr/bin/echo "more command. Verify that each configuration is correct. If they "
/usr/bin/echo "are not correct the appropriate steps must be taken to correct them"
/usr/bin/echo "and new standard files should be created"
/usr/bin/echo ""
/usr/bin/echo "Sleeping for 8 seconds"
/usr/bin/echo ""
/usr/bin/sleep 8

if [ -s ${LVLNBOOT_STANDARD} ] ;
then
/usr/bin/echo "============  VERIFY LVLNBOOT CONFIGURATION
======================="
/usr/bin/sleep 1
/usr/bin/echo ""
/usr/bin/more ${LVLNBOOT_STANDARD}
/usr/bin/echo ""
/usr/bin/echo ""
else
/usr/bin/echo ""
/usr/bin/echo "LVLNBOOT Configuration Standard was not successfully created, CHECK IT."
/usr/bin/sleep 1
/usr/bin/echo ""
fi

#############################################################################

'compare_standard')

# verify existance of standards file, if it does not exist create it and email.
# Compare current config to standard. If differences are found # email admin team.
HOSTID=`hostname`
```

```
DATE=`date '+%y%m%d'`
CONFIGPATH=/usr/local/bin/security
LVLNBOOT_STANDARD=${CONFIGPATH}/.LVLNBOOT.STANDARD
ALL_STANDARD_HIST=${CONFIGPATH}/.ALL.STANDARD.HIST
LVLNBOOT_COMPARE=${CONFIGPATH}/lvlnboot.compare.${DATE}
LVLNBOOT_DIFF=${CONFIGPATH}/lvlnboot.diff.${DATE}

if [ -s ${LVLNBOOT_STANDARD} ] ;
then
# compare current lvlnboot to standard.
        /usr/sbin/lvlnboot -v | /usr/bin/grep -v alternate > ${LVLNBOOT_COMPARE}
        /usr/bin/chmod 600 ${LVLNBOOT_COMPARE}
        /usr/bin/diff ${LVLNBOOT_STANDARD} ${LVLNBOOT_COMPARE} >
${LVLNBOOT_DIFF}
        /usr/bin/chmod 600 ${LVLNBOOT_DIFF}
        if [ -s ${LVLNBOOT_DIFF} ] ;
        then
        /usr/bin/echo "###############" >> ${LVLNBOOT_DIFF}
        /usr/bin/echo "This email was created by /usr/local/bin/monitor_config" >> ${LVLNBOOT_DIFF}
        /usr/bin/echo "The lvlnboot configuration has changed, CHECK IT IMMEDIATELY" >>
${LVLNBOOT_DIFF}
        /usr/bin/echo "###############" >> ${LVLNBOOT_DIFF}
        /usr/bin/mailx -s "SECURITY: monitor_config/lvlnboot output from $HOSTID" INSERT EMAIL
ADDRESSES HERE < ${LVLNBOOT_DIFF}
        else
        echo " "
        fi
else
#standard did not exist and compare was attmepted, create standard and email notice.
DATE=`date '+%c'`
LVLNBOOT_EMAIL=${CONFIGPATH}/lvlnboot.email
        /usr/sbin/lvlnboot -v | /usr/bin/grep -v alternate > ${LVLNBOOT_STANDARD}
        /usr/bin/chmod 600 ${LVLNBOOT_STANDARD}
        /usr/bin/echo "New LVLNBOOT Standard Created on $DATE" >> ${ALL_STANDARD_HIST}
        /usr/bin/echo "The LVLNBOOT.STANDARD file did not exist and a " >
${LVLNBOOT_EMAIL}
        /usr/bin/echo "compare_standard was attempted. This email was " >> ${LVLNBOOT_EMAIL}
        /usr/bin/echo "created by the /usr/local/bin/monitor_config " >> ${LVLNBOOT_EMAIL}
        /usr/bin/echo "script. " >> ${LVLNBOOT_EMAIL}
        /usr/bin/mailx -s "Lvlnboot Standard file did not exist on $HOSTID created it" INSERT EMAIL
ADDRESSES HERE < ${LVLNBOOT_EMAIL}
        /usr/bin/rm ${LVLNBOOT_EMAIL}
fi

        ;;

##########################################################################################
###

*)
        echo "usage: $0 {create_standard|compare_standard}"
        rval=1
        ;;
esac
```

```
exit $rval
```

Appendix F – Monitor_passwd script

```
#!/bin/ksh
#
# This script will monitor the /etc/passwd file on each system ( it will also
# monitor the NIS password map). The script will
# specifically look for account entries where the password field is null.
# If accounts are found with no password the username and a short message and
# saved to a file in /usr/local/bin/security and the administration team is
# notified via email.
#
#
#
#
HOST=`/usr/bin/hostname`
NO_PASSWD_YP=/usr/local/bin/security/passwd.yp.none
NO_PASSWD=/usr/local/bin/security/passwd.none
PASSWD_EMAIL=/usr/local/bin/security/password.email
#
# if script is executing on nis master check both passwd files
#
if [ "$HOST" = "NIS_MASTER" ] ;
then
#
        if [ -s ${NO_PASSWD_YP} ] ;
        then
        /usr/bin/cat /dev/null > ${NO_PASSWD_YP}
        fi
        /usr/bin/touch ${NO_PASSWD_YP}
        /usr/bin/chmod 600 ${NO_PASSWD_YP}
#
#
        if [ -s ${NO_PASSWD} ] ;
        then
        /usr/bin/cat /dev/null > ${NO_PASSWD}
        fi
        /usr/bin/touch ${NO_PASSWD}
        /usr/bin/chmod 600 ${NO_PASSWD}
#
        /usr/bin/ypcat passwd | /usr/bin/awk '{ FS=":"; print $1 " " $2 }' |
        while read username_yp password_yp ;
        do
        RESULT_PASSWD_YP=${password_yp:-1234}
        if [ "$RESULT_PASSWD_YP" = "1234" ] ;
        then
        /usr/bin/echo "The " $username_yp " account has no password in passwd.yp" >>
${NO_PASSWD_YP}
        fi
        done
#
#
        /usr/bin/cat /etc/passwd | /usr/bin/grep -v "+::0:0:::" | /usr/bin/awk '{ FS=":"; print $1 " " $2 }' |
```

```
        while read username password ;
        do
        RESULT_PASSWD=${password:-1234}
        if [ "$RESULT_PASSWD" = "1234" ] ;
        then
        /usr/bin/echo "The " $username " account has no password in /etc/passwd" >> ${NO_PASSWD}
        fi
        done
#
#
        if [ -s ${NO_PASSWD_YP} ] ;
        then
        /usr/bin/echo "Account(s) exist in the password map w/out passwords on " $HOST". See the file"
>> ${PASSWD_EMAIL}
        /usr/bin/echo "/usr/local/bin/security/password.yp.none on " $HOST " for specific information" >>
${PASSWD_EMAIL}
        /usr/bin/mailx -s "SECURITY: Account(s) exist in password map w/out passwords" INSERT
EMAIL ADDRESSES HERE < ${PASSWD_EMAIL}
        /usr/bin/rm ${PASSWD_EMAIL}
        fi
#
#
        if [ -s ${NO_PASSWD} ] ;
        then
        /usr/bin/echo "Account(s) exist in the password file w/out passwords on " $HOST ". See the file"
>> ${PASSWD_EMAIL}
        /usr/bin/echo "/usr/local/bin/security/password.none on " $HOST " for specific information" >>
${PASSWD_EMAIL}
        /usr/bin/mailx -s "SECURITY: Account(s) exist in password map w/out passwords" INSERT
EMAIL ADDRESSES HERE < ${PASSWD_EMAIL}
        /usr/bin/rm ${PASSWD_EMAIL}
        fi
#
#
# if script is executing on any system but nis master check local passwd file only.
else
#
#
        if [ -s ${NO_PASSWD} ] ;
        then
        /usr/bin/cat /dev/null > ${NO_PASSWD}
        fi
        /usr/bin/touch ${NO_PASSWD}
        /usr/bin/chmod 600 ${NO_PASSWD}
#
#
        /usr/bin/cat /etc/passwd | /usr/bin/grep -v "+::0:0:::" | /usr/bin/awk '{ FS=":"; print $1 " " $2 }' |
        while read username password ;
        do
        RESULT_PASSWD=${password:-1234}
        if [ "$RESULT_PASSWD" = "1234" ] ;
        then
        /usr/bin/echo "The " $username " account has no password in /etc/passwd" >> ${NO_PASSWD}
        fi
        done
```

```
#
#
        if [ -s ${NO_PASSWD} ] ;
        then
        /usr/bin/echo "Account(s) exist in the password file w/out passwords on " $HOST ". See the file"
>> ${PASSWD_EMAIL}
        /usr/bin/echo "/usr/local/bin/security/password.none on " $HOST " for specific information" >>
${PASSWD_EMAIL}
        /usr/bin/mailx -s "SECURITY: Account(s) exist in password map w/out passwords" INSERT
EMAIL ADDRESSES HERE < ${PASSWD_EMAIL}
        /usr/bin/rm ${PASSWD_EMAIL}
        fi
#
#
fi
```

Appendix G – Tar_it Script

```
#!/bin/ksh
#
# Created by Paul Yarborough
# This script will tar contents from $SOURCE to $DEST.  Where $SOURCE is the source directory and
$DEST is the
# destination directory. First $SOURCE and $DEST will be checked to see if they exist.
# Allowed exit values:
#       0 = success
#       1 = failure

PATH=/usr/sbin:/usr/bin:/sbin
export PATH

rval=0

# Check to see that source directory and destination directory exist.

set_return() {
        x=$?
        if [ $x -ne 0 ]; then
          echo "SCRIPT FAILED: $x"
          rval=1    # script FAILed
        fi
}

echo "Enter source directory: \c"; read SOURCE
echo $SOURCE

if [ -d $SOURCE ] ;
  then
        touch $SOURCE
  else
        echo "ERROR: SOURCE DIRECTORY DOESN'T EXIST"
           rval=1  # script FAILed
fi
```

```
echo "Enter destination directory: \c"; read DEST
echo $DEST
if [ -d $DEST ] ;
  then
        cd $SOURCE
     tar cvpf - . | ( cd $DEST ; tar xvpf - )
  else
        echo "ERROR: DESTINATION DIRECTORY DOESN'T EXIST"
          rval=1  # script FAILed
fi
```

Appendix H – System Status Diary

TCI Information Resources - Finance

System Administration Status 02/23/1998

MISC TASKS:

1-Nsswitch.conf , Lets make them all the same.
COMPLETED 02/12/98

2-Need to do a security procedures audit on all systems and define security procedures.
This is a work in progress. Meeting 02/24 at 10:00 A.M.

3-Have we ever completed the load balancing of the openvision schedules, NO. need to look at this in the near future. Maybe andrea could help us collect this information.
We also need to review all backups and create docs for each server.

4-DNS cache server
Glenn is working on it 02/09
It is now an unofficial secondary DNS server for the Finance subnet.
COMPLETED
Need to add the resolv.conf changes to Install doc.

5-/etc/ntp.conf has been modified on all. Need to make sure this change is added to Install document.

6-Get a sign made for HP CE's. What log books are we missing.

7-T600 upgrade question HSC hardware path changes HP ref num a4921549
Here is an example of how the hardware paths will change when using hsc daughter cards. 2/7 = 2/28
2/11 = 2/44
4/28
4/44 etc.
This call has been turned over to software team to determine if device names change.

8-The 64 bit contrib tools are on merced in /var/spool/sw/64BIT_CONTRIB_TOOLS they need to be put on all 64 bit machines in /usr/contrib/bin/?????? This is not a depot.

9-DES Ram disk bad fuse or power supply???? Bad fuse Glenn will pick one up.
Has not been able to find fuse yet 01/15 250V 5A fuse ??????Call DES

10-Need to call HP "Some Processes wouldn't die" HP ref num A4988622 01/05 They are looking into it. They are faxing me some information on oracle processess that will not die. Problem could also be zombie, defunct or processes that have no parent. I will monitor this call has been closed.
MONITOR AT NEXT REBOOTS
New call ID A5029102 logged 01/29 emailed process list. They are looking into it.
Need to boot single user mode shutdown -0 y, create a list of processes, copy rc.log and list out rc?.d to a file. email all to hp

11-Remove patch PHNE_12576 from all per HP, replace it with PHNE_12872.
COMPLETED

12-apply patches_6 to all.
COMPLETED

13-Need to move workstations to 6th floor. All users will be notified and after move logins/NIS will be tested.

14-Need to look at DES drives, how many can we free up? Louis wants to sell.

15-Support tools are not currently installed on

16- We may want to consider adjusting the time out on certain devices throughout the environment. pvchange -t 180 /dev/dsk/….. This may limit the occurrence of bogus LVM powerfail errors in syslog/dmesg. Don't do it yet.

17-Need to add a list of default kernel drivers and tunable parameters to install doc that we will use for all 10.20.

18-Where are all the packingo slips for hardware that we have rec. ie FC hardware.

19-Version 8 of diags is PHSS_12285

20-Test mirror_stale script when mirrors are syncing on a machine.
COMPLETED

21-Need to create a script similar to sys_config to monitor configs ie lvlnboot, swapinfo????

22-Need to remove PHNE_13669 from all and replace with PHNE_12428 if necessary.
COMPLETED

23-changed /etc/nettlgen.conf to fix networking timeouts
Change it on all. Edit the above file at line "NFS" change 12 to 8 and then stop and restart nettl.
COMPLETED
need to add this to install doc.

24-Problem with double mirror config. HP call ID a5055885
COMPLETED

25-Mirrors are split on ……. Resync 02/16 in evening.
COMPLETED

26-The patch PHKL_13912 has been recalled it is in patches_7..
Remove and replace with PHKL_14127 at next downtime.

3=27-Call a5061414 problem with cde on standalone bbox. Copy /usr/dt/config/Xconfig to /etc/dt/config/Xconfig the uncomment the following line.
Dtlogin*authorize: False
Stop and restart cde ie init2 then init3
COMPLETED

SERVER_A:

01/27/98 - Swap G out for K, have 3 days
01/29/98 - we are done except we need to verify config.
COMPLETED

02/04/97- G box returned to HP
COMPLETED

02/07/98 - Attempted to configure triple mirror and additional disk. Problems, trying to get downtime
02/13/97 to continue. Need to fix vg00 first, thinks it has 9 devices. COMPLETED.

02/14/98 - 12 hours scheduled downtime.

SERVER_B:

01/13/98 - LPMC keep an eye on it.

01/13/98 - 2411D0007 Predictive reported bad sectors? On 0/28/20.12 = c17t12d0 = vg08/lvol1 NO OS errors being logged but replace disk today at 16:00

01/14/98 - 2411t0168 Predictive called again. 8/28/12.8 , 6/28/12.12, 4/28/44.2 , 4/28/28.15, 2/28/52.12 all replaced.

02/14/98- 2 hours scheduled downtime.

02/19/98 - LPMC slot 13 module 0 I-Cache, keep an eye on it.

SERVER_C:

11/18/97 - Reboot after Panic: Data Page Fault. Glenn is working. HP ref num a4929159
paniced again 11/19 Escalation number 2411p5678 system was patched see patches_5 in misc above.
HP is doing further analysis on core files should be hearing back week of 12/01 ??????
HP recommended PHKL_11741 in addition to prev. recommended need to download and add to depot.
COMPLETED
1 hour unscheduled downtime
SERVER_D:

02/12/98 - HP num 2411T1342 lost power supply in hotel 2/52 replaced it, 2 hours downtime.

02/14/98 - 8 hours scheduled downtime.

SERVER_E:

12/17/97 - now a NIS slave.

COMPLETED

01/24/98 - Modem is hung therefore predictive is not working. Reboot at dev downtime 01/24 and test. Can't get downtime have to wait until 01/31
COMPLETED 02/07 predictive is working.

01/29/98 - rev of installed FDDI software is behind. Need to reinstall. Or is patching enough????

02/15/98 - Problem with NIKE array, amber lights on 4 mechs, had to power cycle. Keep an eye on it. Affected vg22. Down for 45 min.

02/20/98 - Lost fan in one of the AutoRAID's unit overtemped and shutdown. HP did not have a fan so one was shipped overnight. We ran with cabinet doors overnight and replaced fan. REF #2411T1545 1 hour downtime.