LASER INTERFEROMETER GRAVITATIONAL WAVE OBSERVATORY
-LIGO-
CALIFORNIA INSTITUTE OF TECHNOLOGY
MASSACHUSETTS INSTITUTE OF TECHNOLOGY

Technical Note **LIGO-T030292-00-C** 12/15/03

ROBO BOOTLOGGER

Chethan Parameswariah

This is an internal working note
of the LIGO Project.

LIGO Livingston Observatory
19100 Ligo Lane
Livingston, LA 70754
Phone (225) 686-3100
Fax (225) 686-7189

LIGO Hanford Observatory
Route 10, Mile Marker 2
Richland, WA 99352-0159
Phone (509) 372-8106
Fax (509) 372-8137

California Institute of Technology
LIGO Project – MS 51-33
Pasadena CA 91125
Phone (626) 395-2129
Fax (626) 304-9834
E-mail: info@ligo.caltech.edu

Massachusetts Institute of Technology
LIGO Project – MS 20B-145
Cambridge, MA 01239
Phone (617) 253-4824
Fax (617) 253-7014
E-mail: info@ligo.mit.edu

WWW: http://www.ligo.caltech.edu

1. ABSTRACT

This document describes the working of the automatic web logging system called "ROBO BOOTLOGGER" for logging reboots of the VME processors and sun workstations present on the Control and Data Systems (CDS) network, logging of any vacuum system epics changes such as button presses, slider moves, text entry changes etc., and logging any filter changes to the interferometer's various control systems such as length sensing control system (LSC), alignment sensing control (ASC) and the suspension control systems (SUS), to the existing e-log (web log) in the detector group. Robo-bootlogger is part of the collection of software robots - "*SOFT-ROBOTS*" now working at LLO to ease and improve efficiency.

2. INTRODUCTION

Automation of LLO CDS Systems is essential and critical to management of CDS Network with the advancement of LLO interferometer into low noise commissioning and science runs.

Data from the epics and front-end processor reboots, the sun workstation reboots, the vacuum changes and the filter changes to the interferometer control systems are collated and presented at one central location for easy retrieval, analysis and troubleshooting. Data collected and logged is useful for forensic data analysis and tracking of problems.

ROBO BOOTLOGGER is a software robot that collects 1) essential reboot data from various vme processors and sun workstations used for data acquisition and control of the interferometer, 2) the vacuum epics changes, and 3) the interferometer control system filter changes. The collected data is collated and logged on the e-log every day.

3. OPERATION

ROBO PLOTTER is part of "*SOFT-ROBOTS*" - a collection of software programs and scripts called '*software robots'* that are run at LIGO to automate various duties of the scimons, operators and engineers to ease and improve efficiency while maintaining reliability and consistency on a daily basis.

ROBO BOOTLOGGER itself is a collection of simple programs and scripts that are run in tandem to collect data i.e., *time and name of processor,* when a processor or a sun workstation is rebooted or started after a shutdown, *time, channel name and value* for vacuum epics value changes, and *time and filter file changed,* for any filter changes.

The main program "autoelog_reboots.pl" is a data collection and auto e-logger program. This is a Perl program that runs as a cronjob every morning at 2 minutes past midnight and collects the data for the previous day from 00:00 hrs to 23:59 hrs and e-logs the collected data on the page corresponding to previous day's e-log entries. This cronjob is on control3 in the mass storage room and is run as controls.

The program listing for the autoelog_reboots.pl is in Appendix.

The perl program relies on other programs to create log files for the reboots, vacuum changes and the filter changes.

1) EPICS reboots: The epics processors have a standard epics supplied iocLogserver which is run on the host machine – LLO1 at LLO. The iocLogserver records all transactions such as reboots, error messages and warnings from an epics ioc processor. The iocLogserver saves these messages in the log file */cvs/cds/llo/logs/epics.log.* The autoelog_reboots.pl parses this log file and grabs the list of reboots done on the previous day to elog.

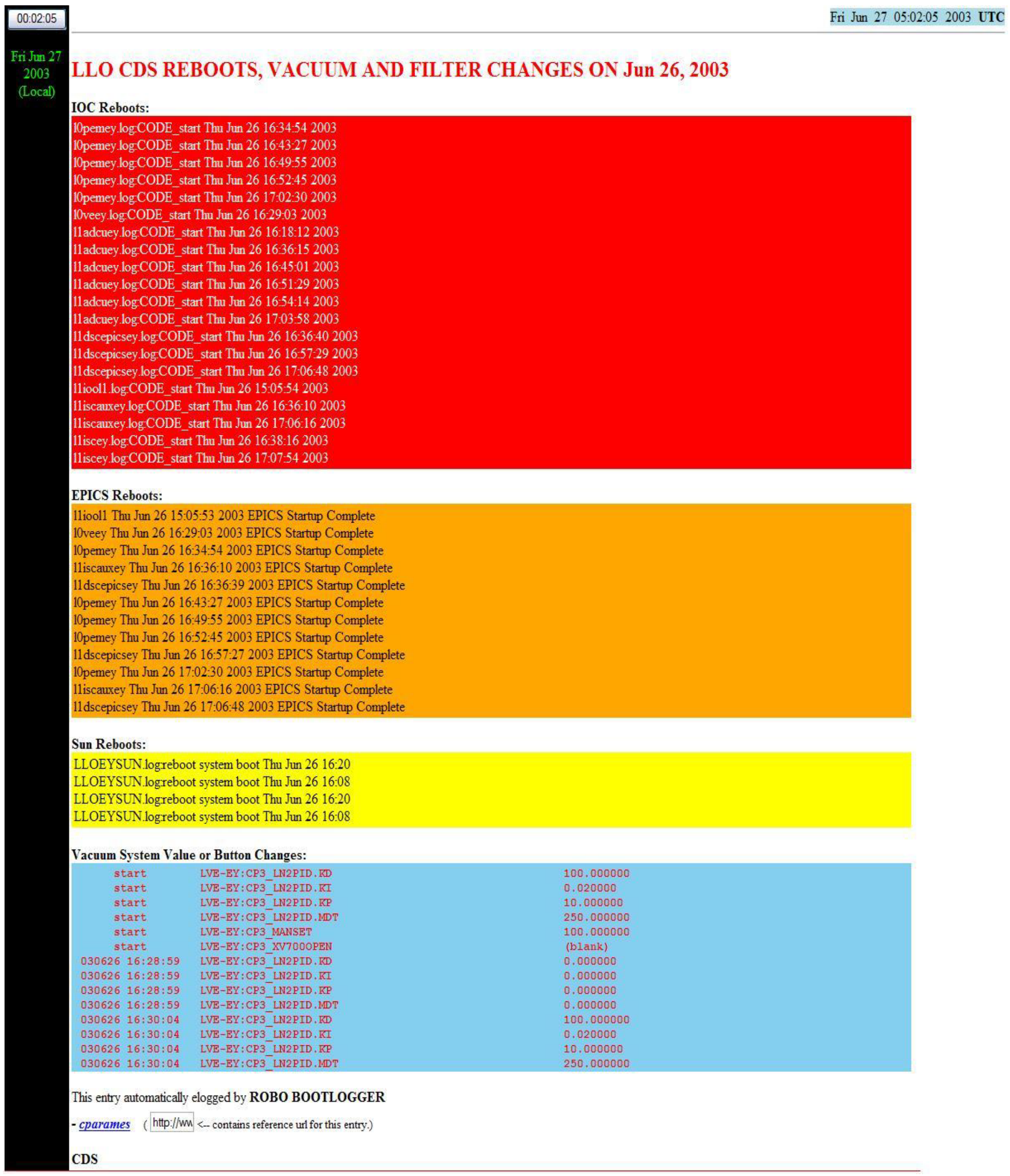
2) Front End reboots: All processors whether epics or non-epics have a small script that is included in the startup.cmd file. When the startup.cmd file is loaded on to the processors, the script creates a reboot.log file in the target area of each ioc. A cronjob process running on LLO1 looks for these reboot.log files every second and logs the reboots in their respective ioc directory in the */cvs/cds/project/targets/* directory. The autoelog_reboots.pl program then finds the current and latest reboots and elogs them.

3) Sun Reboots: All Sun machines at LLO are also scanned for their reboot messages on them by a cronjob running at LLO and logged into a file /cvs/cds/llo/logs/sunreboot.log. This file is then read by the robo-bootlogger and elogged.

4) Vacuum changes: All vacuum changes are obtained from the conlog (a program written by Peter Shawhan at Caltech) for the previous day and elogged.

5) Filter changes: The filter changes are saved in the /cvs/cds/llo/chans/filter_archive directory for each sub system. All the previous day's filter changes are then read by the autoelog_reboots.pl program, to be elogged.

The gif image of the automatic elog done by ROBO-BOOTLOGGER is shown in figure 1. Filter changes are also elogged with a dark blue background but not shown in the picture.



LLO CDS REBOOTS, VACUUM AND FILTER CHANGES ON Jun 26, 2003

IOC Reboots:

```
l0pemey.log:CODE_start Thu Jun 26 16:34:54 2003
l0pemey.log:CODE_start Thu Jun 26 16:43:27 2003
l0pemey.log:CODE_start Thu Jun 26 16:49:55 2003
l0pemey.log:CODE_start Thu Jun 26 16:52:45 2003
l0pemey.log:CODE_start Thu Jun 26 17:02:30 2003
l0veey.log:CODE_start Thu Jun 26 16:29:03 2003
l1adcuey.log:CODE_start Thu Jun 26 16:18:12 2003
l1adcuey.log:CODE_start Thu Jun 26 16:36:15 2003
l1adcuey.log:CODE_start Thu Jun 26 16:45:01 2003
l1adcuey.log:CODE_start Thu Jun 26 16:51:29 2003
l1adcuey.log:CODE_start Thu Jun 26 16:54:14 2003
l1adcuey.log:CODE_start Thu Jun 26 17:03:58 2003
l1dscepicsey.log:CODE_start Thu Jun 26 16:36:40 2003
l1dscepicsey.log:CODE_start Thu Jun 26 16:57:29 2003
l1dscepicsey.log:CODE_start Thu Jun 26 17:06:48 2003
l1iool1.log:CODE_start Thu Jun 26 15:05:54 2003
l1iscauxey.log:CODE_start Thu Jun 26 16:36:10 2003
l1iscauxey.log:CODE_start Thu Jun 26 17:06:16 2003
l1iscey.log:CODE_start Thu Jun 26 16:38:16 2003
l1iscey.log:CODE_start Thu Jun 26 17:07:54 2003
```

EPICS Reboots:

```
l1iool1 Thu Jun 26 15:05:53 2003 EPICS Startup Complete
l0veey Thu Jun 26 16:29:03 2003 EPICS Startup Complete
l0pemey Thu Jun 26 16:34:54 2003 EPICS Startup Complete
l1iscauxey Thu Jun 26 16:36:10 2003 EPICS Startup Complete
l1dscepicsey Thu Jun 26 16:36:39 2003 EPICS Startup Complete
l0pemey Thu Jun 26 16:43:27 2003 EPICS Startup Complete
l0pemey Thu Jun 26 16:49:55 2003 EPICS Startup Complete
l0pemey Thu Jun 26 16:52:45 2003 EPICS Startup Complete
l1dscepicsey Thu Jun 26 16:57:27 2003 EPICS Startup Complete
l0pemey Thu Jun 26 17:02:30 2003 EPICS Startup Complete
l1iscauxey Thu Jun 26 17:06:16 2003 EPICS Startup Complete
l1dscepicsey Thu Jun 26 17:06:48 2003 EPICS Startup Complete
```

Sun Reboots:

```
LLOEYSUN.log:reboot system boot Thu Jun 26 16:20
LLOEYSUN.log:reboot system boot Thu Jun 26 16:08
LLOEYSUN.log:reboot system boot Thu Jun 26 16:20
LLOEYSUN.log:reboot system boot Thu Jun 26 16:08
```

Vacuum System Value or Button Changes:

```
     start        LVE-EY:CP3_LN2PID.KD                          100.000000
     start        LVE-EY:CP3_LN2PID.KI                          0.020000
     start        LVE-EY:CP3_LN2PID.KP                          10.000000
     start        LVE-EY:CP3_LN2PID.MDT                         250.000000
     start        LVE-EY:CP3_MANSET                             100.000000
     start        LVE-EY:CP3_XV700OPEN                          (blank)
 030626 16:28:59  LVE-EY:CP3_LN2PID.KD                          0.000000
 030626 16:28:59  LVE-EY:CP3_LN2PID.KI                          0.000000
 030626 16:28:59  LVE-EY:CP3_LN2PID.KP                          0.000000
 030626 16:28:59  LVE-EY:CP3_LN2PID.MDT                         0.000000
 030626 16:30:04  LVE-EY:CP3_LN2PID.KD                          100.000000
 030626 16:30:04  LVE-EY:CP3_LN2PID.KI                          0.020000
 030626 16:30:04  LVE-EY:CP3_LN2PID.KP                          10.000000
 030626 16:30:04  LVE-EY:CP3_LN2PID.MDT                         250.000000
```

This entry automatically elogged by **ROBO BOOTLOGGER**

- *cparames* (http://ww <-- contains reference url for this entry.)

CDS

Figure 1: ROBO-BOOTLOGGER's elog entry on 26 Jun 2003.

4. CONCLUSION

The ROBO-BOOTLOGGER has been successfully working at LLO for almost a year now and is helpful in diagnosing and catching reboots that are not elogged. It helps to also identify and keep a log of vacuum changes. The logs in the elog are tabulated and color-coded. The heading of the elog entry made shows which sections are elogged by listing only those in the top most heading. Further work on fine-tuning the robot is underway.

APPENDIX

1) autoelog_reboots.pl – Automatic reboots, vacuum and filter changes elog

```
#!/opt/apps/perl_5.6.1/bin/perl
#
# autoelog_reboots.pl  Chethan Parameswariah First release May 22
2003
#
# ###################
#
# May 22, 2003          First Release with ioc and epics reboots
# May 28, 2003          Added the vacuum changes to be elogged
# May 30, 2003          Added the filter changes to be elogged
#
#
#
# Need these lib modules - this prepends to @INC at run time
#
use lib "/opt/apps/perl_5.6.1/modules/HTML-Parser-2.22/blib/lib";
use lib "/opt/apps/perl_5.6.1/modules/libwww-perl-5.42/lib";
use lib "/opt/apps/perl_5.6.1/modules/URI-1.02";
use lib "/opt/apps/perl_5.6.1/modules/HTML-Parser-2.22/lib";
use lib "/opt/apps/perl_5.6.1/modules/MIME-Base64-2.11/blib/lib";

# Tell what modules to use
#
use HTTP::Request::Common qw(POST);
use LWP::UserAgent;
use CGI;

# Set variables
#
$DEBUG = 0;

# This variable is intelligently set by the program later.
$AUTO_ELOG = 0;

# These can select which services are elogged.
$ELOG_IOC_REBOOT = 1;
$ELOG_EPICS_REBOOT = 1;
$ELOG_SUN_REBOOT = 1;
$ELOG_VAC_CHANGES = 1;
$ELOG_FILTER_CHANGES = 1;

# These define the directories
$ioc_reboot_logdir = "/cvs/cds/project/targets/llo";
$epics_reboot_logdir = "/cvs/cds/llo/logs";
$sun_reboot_logdir = "/cvs/cds/llo/logs/sunreboot";
$vac_changes_logdir = "/cvs/cds/llo/logs";
$filter_changes_logdir = "/cvs/cds/llo/chans/filter_archive/l1";

# This defines who elogs it
$username = "cparames";
$password = "wave\$";
```

```
# Get hour, today's date, month and year
($HOUR_NUMBER, $DAY_NUMBER, $MONTH_NUMBER, $YEAR_NUMBER) =
(localtime(time)) [2,3,4,5];
$MONTH_STRING = (qw(Jan Feb Mar Apr May Jun Jul Aug Sep Oct Nov
Dec))[(localtime) [4]];
$MONTH_NUMBER += 1;
   $MONTH_NUMBER_x = $MONTH_NUMBER;
if ($MONTH_NUMBER < 10) {
   $MONTH_NUMBER_x = " ".$MONTH_NUMBER;
}
if ($MONTH_NUMBER < 10) {
   $MONTH_NUMBER = "0".$MONTH_NUMBER;
}
   $DAY_NUMBER_x = $DAY_NUMBER;
if ($DAY_NUMBER < 10) {
   $DAY_NUMBER_x = " ".$DAY_NUMBER;
}
if ($DAY_NUMBER < 10) {
   $DAY_NUMBER = "0".$DAY_NUMBER;
}
$YEAR_NUMBER += 1900;

# Get yesterday's date, month and year
($DAY1_NUMBER, $MONTH1_NUMBER, $YEAR1_NUMBER) = (localtime(time-
86400)) [3,4,5];
$MONTH1_STRING = (qw(Jan Feb Mar Apr May Jun Jul Aug Sep Oct Nov
Dec))[(localtime(time-86400)) [4]];
$MONTH1_NUMBER += 1;
   $MONTH1_NUMBER_x = $MONTH1_NUMBER;
if ($MONTH1_NUMBER < 10) {
   $MONTH1_NUMBER_x = " ".$MONTH1_NUMBER;
}
if ($MONTH1_NUMBER < 10) {
   $MONTH1_NUMBER = "0".$MONTH1_NUMBER;
}
   $DAY1_NUMBER_x = $DAY1_NUMBER;
if ($DAY1_NUMBER < 10) {
   $DAY1_NUMBER_x = " ".$DAY1_NUMBER;
}
if ($DAY1_NUMBER < 10) {
   $DAY1_NUMBER = "0".$DAY1_NUMBER;
}
$YEAR1_NUMBER += 1900;

if ($DEBUG) {
print "Hour = $HOUR_NUMBER\tMonth = $MONTH_STRING -
$MONTH_NUMBER\tDay = $DAY_NUMBER,\tYear = $YEAR_NUMBER\n";
print "Yest Hour = $HOUR1_NUMBER\tYest Month = $MONTH1_STRING -
$MONTH1_NUMBER\tYest Day = $DAY1_NUMBER,\tYest Year =
$YEAR1_NUMBER\n";
}

#use this if you want to test the script for a day when reboot
was made, change the value.
#$DAY_NUMBER = 3;
```

```
###### Get the info ######

# Get the ioc reboots - generate a temp file #

# Remove the temp file if it exists
#$command = "if (-e $ioc_reboot_logdir/iocreboot.temp) rm
$ioc_reboot_logdir/iocreboot.temp";
#system($command);
# Generate a temp file for the previous day
$command = "cd $ioc_reboot_logdir; grep \"$MONTH1_STRING
$DAY1_NUMBER_x\" *.log > $ioc_reboot_logdir/iocreboot.temp";
system($command);
if ($DEBUG) {
print "Done creating $ioc_reboot_logdir/iocreboot.temp file \n";
}

# Get the epics reboots logged by iocLogSever #
# Generate a temp file for the previous day
$command = "cd $epics_reboot_logdir; grep \"$MONTH1_STRING
$DAY1_NUMBER_x\" epics.log | grep \"EPICS Startup\" >
$epics_reboot_logdir/epics.temp";
system($command);
if ($DEBUG) {
print "Done creating $epics_reboot_logdir/epics.temp file \n";
}

if ($DAY_NUMBER == 1) {
# Move the epics.log file to epics_old directory
$command = "mv $epics_reboot_log_dir/epics.log
$epics_reboot_log_dir/epics_old/epics.log.$DAY1_number$MONTH1_STR
ING$YEAR1_NUMBER; touch $epics_reboot_log_dir/epics.log; chmod
777 $epics_reboot_log_dir/epics.log ";
system($command);
if ($DEBUG) {
print "Done moving and creating $epics_reboot_logdir/epics.log
file \n";
}
}


# Get the sun reboots - generate a temp file #

# Remove the temp file if it exists
#$command = "if (-e $sun_reboot_logdir/sunreboot.temp) rm
$sun_reboot_logdir/sunreboot.temp";
#system($command);
# Generate a temp file for the previous day
$command = "cd $sun_reboot_logdir; grep \"$MONTH1_STRING
$DAY1_NUMBER_x\" *.log > $sun_reboot_logdir/sunreboot.temp";
system($command);
if ($DEBUG) {
print "Done creating $sun_reboot_logdir/sunreboot.temp file \n";
}

# Get the vacuum changes info - generate a temp file #

# Generate a temp file for the previous day
```

```
$command = "cd /cvs/cds/llo/conlog/bin; conlog +epics +interp
between $YEAR1_NUMBER/$MONTH1_NUMBER/$DAY1_NUMBER
$YEAR_NUMBER/$MONTH_NUMBER/$DAY_NUMBER cst LVE- | grep LVE >
/cvs/cds/llo/logs/vacuumchanges.temp";
system($command);
if ($DEBUG) {
print "Done creating $vac_changes_logdir/vacuumchanges.temp file
\n";
}

# Get the filter changes info - generate a temp file #
$command = "cd $filter_changes_logdir;ls -lR * | grep
\"$MONTH1_STRING $DAY1_NUMBER_x\" | grep txt | awk 'BEGIN{FS=\"
\"} {print \$8   \$9}' > /cvs/cds/llo/logs/filterchanges.temp";
if ($DEBUG) {
print $command." \n";
}
system($command);
if ($DEBUG) {
print "Done creating /cvs/cds/llo/logs/filterchanges.temp file
\n";
}

# Put the above file contents to five array variables : list1
list2 list3 list4 list5
open(IN,"$ioc_reboot_logdir/iocreboot.temp")||die "Cannot open
file $ioc_reboot_logdir\/iocreboot.temp\n";
@list1 = <IN>;
close IN;
open(IN,"$epics_reboot_logdir/epics.temp")||die "Cannot open file
$epics_reboot_logdir\/epics.temp\n";
@list2 = <IN>;
close IN;
open(IN,"$sun_reboot_logdir/sunreboot.temp")||die "Cannot open
file $sun_reboot_logdir\/sunreboot.temp\n";
@list3 = <IN>;
close IN;
open(IN,"$vac_changes_logdir/vacuumchanges.temp")||die "Cannot
open file $vac_changes_logdir\/vacuumchanges.temp\n";
@list4 = <IN>;
close IN;
open(IN,"/cvs/cds/llo/logs/filterchanges.temp")||die "Cannot open
file /cvs/cds/llo/logs/filterchanges.temp\n";
@list5 = <IN>;
close IN;

if ($DEBUG) {
print "@list1 \n @list2 \n @list3 \n @list4 \n @list5 \n";
print "Done copy file contents to arrays \n";
}

$comment_string = "";
if (@list1 != "") {
if ($ELOG_IOC_REBOOT) {
$comment_string = $comment_string."<b>IOC Reboots:</b><TABLE
width=90% bgcolor=\"red\"><TR><TD><font color=\"white\"> @list1
</font></TD></TR></TABLE>";
```

```
$AUTO_ELOG=1;
}
}
if (@list2 != "") {
if ($ELOG_EPICS_REBOOT) {
$comment_string = $comment_string."<P><b>EPICS Reboots:</b><TABLE
width=90% bgcolor=\"orange\"><TR><TD><font color=\"black\">
@list2 </font></TD></TR></TABLE>";
$AUTO_ELOG=1;
}
}
if (@list3 != "") {
if ($ELOG_SUN_REBOOT) {
$comment_string = $comment_string."<P><b>Sun Reboots:</b><TABLE
width=90% bgcolor=\"FFFF00\"><TR><TD><font color=\"black\">
@list3 </font></TD></TR></TABLE>";
$AUTO_ELOG=1;
}
}
$VAC_ELOG=0;
$comment_string_header = "<H2><font color=\"red\">LLO CDS REBOOTS
ON $MONTH1_STRING $DAY1_NUMBER, $YEAR1_NUMBER</font></H2>";
if (@list4 != "") {
if ($ELOG_VAC_CHANGES) {
if ($AUTO_ELOG) {
$comment_string_header = "<H2><font color=\"red\">LLO CDS REBOOTS
AND VACUUM CHANGES ON $MONTH1_STRING $DAY1_NUMBER,
$YEAR1_NUMBER</font></H2>";
} else {
$comment_string_header = "<H2><font color=\"red\">LLO CDS VACUUM
CHANGES ON $MONTH1_STRING $DAY1_NUMBER,
$YEAR1_NUMBER</font></H2>";
}
$comment_string = $comment_string."<P><b>Vacuum System Value or
Button Changes:</b><TABLE width=90%
bgcolor=\"skyblue\"><TR><TD><font color=\"red\"><pre><b> @list4
</b></pre></font></TD></TR></TABLE>";
$AUTO_ELOG=1;
$VAC_ELOG=1;
}
}

if (@list5 != "") {
if ($ELOG_FILTER_CHANGES){
if ($AUTO_ELOG) {
if ($VAC_ELOG) {
$comment_string_header = "<H2><font color=\"red\">LLO CDS
REBOOTS, VACUUM AND FILTER CHANGES ON $MONTH1_STRING
$DAY1_NUMBER, $YEAR1_NUMBER</font></H2>";
} else {
$comment_string_header = "<H2><font color=\"red\">LLO CDS REBOOTS
AND FILTER CHANGES ON $MONTH1_STRING $DAY1_NUMBER,
$YEAR1_NUMBER</font></H2>";
}
} else {
```

```
$comment_string_header = "<H2><font color=\"red\">LLO CDS FILTER
CHANGES ON $MONTH1_STRING $DAY1_NUMBER,
$YEAR1_NUMBER</font></H2>";
}
$comment_string = $comment_string."<P><b>Filter Updates and
Changes:</b><TABLE width=90% bgcolor=\"0000CC\"><TR><TD><font
color=\"white\"><pre><b> @list5
</b></pre></font></TD></TR><TR><TD><font color = white><b>Note:
The above filter updates might have occurred due to the reboot of
the processor. Check the reboots above to make
sure.</b></font></TD></TR></TABLE>";
}
}

# comment this out if you want to elog
#$AUTO_ELOG = 0;
###### Auto Elog ######
if ($AUTO_ELOG) {

# Create a new user agent

$ua = LWP::UserAgent->new();


# Since I need a proxy at LLO, set the proxy here
#
$ua->proxy('http','http://london.ligo-la.caltech.edu:80/');


# set url of elog
#
my $URL = 'http://www.ligo-la.caltech.edu/ilog/pub/ilog.cgi?';

# Elog file date
#
$log_file_date = "$MONTH1_NUMBER/$DAY1_NUMBER/$YEAR1_NUMBER";
#$log_file_date = "12/23/2001";

$comment_string_footer = "<P>This entry automatically elogged by
<b>ROBO BOOTLOGGER </b><BR>";
$comments =
$comment_string_header.$comment_string.$comment_string_footer;

if ($DEBUG) {
      print $comments."\n";
}

# POST it with contents and values
#
my $request = POST $URL,
      Content_Type => 'multipart/form-data',
      Content =>
      [
       group => 'detector',
       task => 'makeEntry',
       log_file_date => $log_file_date,
       comments => $comments,
       keywords => 'CDS',
       priority => 'normal',
       entry_author => 'cparames',
       'submit' => 'Submit Log Entry'
      ];

# Dont forget the authorization
#
$request->authorization_basic($username, $password);

# And finally make the call.
$content = $ua->request($request)->as_string;

if ($DEBUG) {
print $content;
};
}

# Make a entry that tells the program ran successfully.
$statuslogfile = "/cvs/cds/llo/logs/robomamalog.html";

  # finally log we have run
  open(LOG,">$statuslogfile")||die "Cannot open
$statuslogfile\n";
  print LOG "<HTML><BODY>\n";
  print LOG "<H4>Robo-MAMA ran successfully </H4> \n";
  print LOG " <HR>\n";
  print LOG "Completed at  ".returnTimeStamp()."\n";
  print LOG "</BODY></HTML>\n";
  close LOG;

# -----------------------------------------------------------------
-------
#
# Subroutine returnTimeStamp()
#
# Returns the current time stamp as a string
#
# -----------------------------------------------------------------
-------
sub returnTimeStamp {

  @timestamp = localtime(time);
  $thisday = (Sun,Mon,Tue,Wed,Thu,Fri,Sat)[$timestamp[6]];
  $MONTH_NUMBER = $timestamp[6] + 1;
  $thismonth =
(Jan,Feb,Mar,Apr,May,Jun,Jul,Aug,Sep,Oct,Nov,Dec)[$timestamp[4]];
  $MONTH_NUMBER = $timestamp[4] + 1;
  if ($MONTH_NUMBER < 10){
    $MONTH_NUMBER = "0".$MONTH_NUMBER;
  }
  $DAY_NUMBER = $timestamp[3];
  if ($DAY_NUMBER < 10){
    $DAY_NUMBER = "0".$DAY_NUMBER;
  }
  # Y2k stuff
  # Year 2000 defined as $timestamp[5] = 100
```

```
    if($timestamp[5]> 99) {
      $timestamp[5] = $timestamp[5] - 100;
      $thisyear = "200".$timestamp[5];
    } else {
      $thisyear = "19".$timestamp[5];
    }
    $YEAR_NUMBER = $thisyear;
    if ($timestamp[2] < 10) {
      $thishour = "0".$timestamp[2];
    } else {
      $thishour = $timestamp[2];
    }
    if ($timestamp[1] < 10) {
      $thismin = "0".$timestamp[1];
    } else {
      $thismin = $timestamp[1];
    }
    if ($timestamp[0] < 10) {
      $thissec = "0".$timestamp[0];
    } else {
      $thissec = $timestamp[0];
    }

    $thisdate = $timestamp[3];

    return "$thisday$thisdate$thismonth$thisyear-
  $thishour:$thismin:$thissec";


  }
```

**************************** END ******************************