**NIST Laboratory Environment Monitoring and Alert System (LEMAS)**

Documentation for building, usage, and configuration

Michael Braine, [michael.braine@nist.gov](mailto:michael.braine@nist.gov)

April 25, 2017

Revision 1.0, established for LEMASdist v1.11 package

**What is this, and why should I read it?**

This documentation outlines device setup and customization options should a user desire to perform changes to the controls that dictate when a MMS message or email is sent to lab users. It assumes no understanding of Python3 syntax, Python libraries that are used, or Linux commands; however, the reader should be, at a minimum, comfortable on a command line interface and know what a raspberry (razˌberē,ˈrazˌb(ə)rē – an edible soft fruit related to the blackberry, consisting of a cluster of reddish-pink drupelets) looks like. The curator, in the context within, is the current maintainer of all the monitoring devices (primary curator) and could include additional members whom are designated to receive the same messages as the primary curator.

**Table of Contents**

Components required for a new device

Contents of LEMASdist package

Logic for when a message is sent

Installation of LEMAS with LEMASRaspbian image

Manual installation of LEMAS

Device customization

**Components required for a new device**

<components list>

<picture of setup>

Using a SDXC microSD card (typically larger than 16 GB capacity, but includes some 16 GB cards) will require exFAT32 formatting.

The current revision of LEMASrun.py was tested to work with the above components, and this document written with the assumption that the device will be using the above parts list. A Raspberry Pi 3 Model B (hereby RPi for the rest of the text) is not required, but knowledge of Python3 (or Python2, if you prefer that flavor) and the native OS will be required for setup, especially if running the program on boot is desired.

**Contents of LEMASdist package**

*Filename Description Raspbian Install Location*

LEMASRun.py Python3 script for logging T3311, alerts /home/pi/LEMASdist

LEMASRunQuiet.py Python3 script for logging T3311, no alerts /home/pi/LEMASdist

LEMASRunLoud.py Python3 script for logging T3311, alerts /home/pi/LEMASdist

NoContact.list list of names to not send messages to /home/pi/LEMASdist

LabSettings.py user-configurable settings file /home/pi/LEMASdist

LabID.py identifies the lab that the device resides /home/pi/LEMASdist

testmsgdate.csv date to send test message to users /home/pi/LEMASdist

Contacts.py(.gpg) GNU\_gpg encrypted, contacts list /home/pi/LEMASdist

Tcontrols.py temperature controls for each lab /home/pi/LEMASdist

Rhcontrols.py humidity controls for each lab /home/pi/LEMASdist

corrections.py corrections for the sensors /home/pi/LEMASdist

version details of the installed LEMAS version /home/pi/LEMASdist

launch-LEMAS executable Bash script to launch system /home/pi/Desktop

config.txt configuration for Adafruit display /boot

lightdm.conf configuration for disabling screensaver /etc/lightdm

**Logic for when a message is sent**

A message is sent from any single device under one of three conditions:

1. When the set environment has exceeded limits, with a graph
2. When the set environment has returned within limits for a period of time, with a graph
3. When the set environment has exceeded limits, and the environment (T or RH) changes in increments of TincSet or RHincSet (LabSettings.py) until the environment returns within limits for a period of time, with a graph

In the first case, the device will send a message with an attached graph to its laboratory owners should temperature or humidity limits be exceeded. These limits can be easily changed, detailed in section *Configuring the device for a lab with existing settings*.

In the second case, the device will send a message with an attached graph to its laboratory owners should temperature or humidity conditions return within limits after a period of time. These limits are the same limits that decide when a message is sent under the third condition and can be easily changed, detailed in section *Configuring the device for a lab with existing settings*.

In the third case, when the environment is outside of the setpoints, the device will send a message for each occasion the environment changes by TincSet or RhincSet. E.g., the temperature upper limit is 25 deg. C, and TincSet is 5 deg. C, there will be a message generated at 30 deg. C, 35, deg. C, etc, until the temperature returns below 25 deg. C.

**Installation of LEMAS with LEMASRaspbian image**

The LEMASRaspbian image is a copy of the entire ext4 filesystem of a working LEMAS device. The Raspbian kernel is configured and compiled to be securely used on the NIST network. LEMAS is already installed in the image.

1. With a microSD card writer, copy the latest image onto a blank microSD card.
2. Upon supplying power to the Raspberry Pi, the OS will load and LEMAS will launch by itself.

**Manual installation of LEMAS**

**Device setup from a blank microSD card**

Required to install Raspbian to a microSD card is a microSD card reader, or adapter for SD-to-microSD.

*Installing Raspbian OS from a Windows Machine*

1. Navigate a web browser to https://www.raspberrypi.org/downloads/raspbian/ and download the latest full Raspbian ZIP image for Raspberry Pi 3.
2. Navigate a web browser to https://etcher.io, download and install the latest Etcher release.
3. Insert the microSD card into the computer. Launch etcher. Select the downloaded Raspbian image file, select the drive containing the microSD card, select Flash!.
4. Once Raspbian has been written to the microSD card and verified by Etcher, remove the microSD card from the computer and insert it into the Raspberry Pi. Connect the Raspberry Pi to a computer monitor (*NOT THE ADAFRUIT DISPLAY*, the RPi must be configured because the display does not have onboard video codec feedback. Running the installer BASH will configure the Adafruit display). Supply power to the Raspberry Pi to through the micro USB cable. The RPi will boot directly into the default user “pi” with default password “raspberry”.

**Updating Raspbian OS and Python3**

1. Activate the US-based keyboard layout by navigating through the start menu (raspberry icon > Preferences > Mouse and Keyboard). Click the Keyboard tab, click on the Keyboard Layout button, select United States under Country, and English (US) under Variant.
2. Connect the device to the internet with Ethernet and use the network icon on the taskbar to connect to the NIST Research Network. Ask the OU IT Security Officer or group IT Security Representative for assistance if needed. Do note that you will need access to repositories external to NIST at least once when setting up the REN connection. Alternatively, the NIST Guest WiFi is suitable for updates, since the Raspberry Pi 3 Model B has a built-in WiFi chip. Similarly, use the network icon to connect to NIST Guest WiFi.
3. In Raspbian, open a terminal either by clicking the button on the taskbar or navigating through the start menu (raspberry icon > Accessories > Terminal). Type the following command in the terminal and press enter:   
    sudo apt-get update   
   This command will ask for a list of updates of all installed packages that are managed by the apt-get package manager, including Raspbian OS and Python 3. Once the list has been updated, type the following command in the terminal and press enter:   
    sudo apt-get upgrade   
   This command will download and install the updates. Reply y to any y/n prompts.
4. (optional, for freeing extra disk space)In the terminal, enter:   
    sudo apt-get remove minecraft-pi -y && sudo apt-get remove bluej -y && sudo apt-get remove geany -y && sudo apt-get remove greenfoot -y && sudo apt-get remove nodered -y && sudo apt-get remove scratch -y && sudo apt-get remove scratch2 -y && sudo apt-get remove sense-hat -y && sudo apt-get remove sonic-pi -y   
   These commands will remove unused programs preloaded on Raspbian.
5. In the terminal, enter:   
    sudo apt-get autoclean   
   This command will clean up any files that may no longer be used due to the update processes.
6. In the terminal, enter:   
    sudo apt-get install python3-pyserial python3-numpy python3-matplotlib   
   This command installs the PySerial, NumPy, and matplotlib libraries for Python3 for serial interfacing, numerical math operations similar to Matlab, and matrix plotting library, respectively. Note that the python3 designation here is not a Linux command to run Python, it is asking for the libraries in the repositories to download that are built for Python3.
7. In the terminal, enter:   
    sudo python3 /usr/bin/pip install minimalmodbus   
   This will install the minimalmodbus library for Modbus RTU communications with Python’s pip installer for Python 3.

**Locally compiling custom Raspbian Image with NIST-required security features**

The Raspbian kernel must have additional features enabled to be allowed on the NIST network. For reference, the instructions for custom building a Raspbian kernel is found on the Raspberry Pi website (<https://www.raspberrypi.org/documentation/linux/kernel/building.md>). The kernel should be built on a Raspberry Pi 3 for the reason that weird behaviors were noted to occur when cross-compiled with a Ubuntu 14 system.   
An internet connection is required to grab the OS source code and build dependencies.

Open a terminal with CTRL+ALT+T. Install git and build dependencies using:   
 sudo apt install git bc

In the terminal, obtain the source code using:   
 git clone –depth=1 https://github.com/raspberrypi/linux

Compute the default build configuration:   
cd linux   
KERNEL=kernel7   
make bcm2709\_defconfig

Use make-menuconfig to configure the kernel to enable CONFIG\_CPU\_SW\_DOMAIN and CONFIG\_AUDIT. Enter make-manuconfig in the terminal to launch menuconfig, and search for the required features in the kernel menu. Save when finished.

Build and install the kernel using:   
make -j4 zImage modules dtbs   
sudo make modules\_install   
sudo cp arch/arm/boot/dts/\*.dtb /boot/   
sudo cp arch/arm/boot/dts/overlays/\*.dtb\* /boot/overlays   
sudo cp arch/arm/boot/zImage /boot/$KERNEL.img

Upon rebooting, the Raspberry Pi will use the new, configured kernel.

**Configuring the Raspberry Pi 3**

1. On a flash drive, load the LEMASdist folder provided by the curator. Plug the flash drive into the RPi on any USB slot. Open file manager with the folder icon on the taskbar, or navigating through the start menu (raspberry icon > Accessories > File Manager).
2. If the dropdown menu does not already read “Places”, click the dropdown that reads “Directory Tree” and change to “Places”. Access the flash drive and copy/paste the LEMASdist folder onto the desktop.
3. In the terminal, use the copy command (cp) to move the config.txt file to /boot/:   
    sudo cp home/pi/Desktop/LEMASdist/config.txt /boot/config.txt   
   The config.txt file will allow the Adafruit display to work with the RPi upon restarting with the display connected and powered. If tilde (~) does not appear, rather another symbol, then the keyboard needs changed to English (US) (see Step 9).
4. In the terminal, enter the following command to change audio output from over HDMI to the 3.5mm jack. This will prevent potential issues when using the Adafruit display:   
    sudo amixer cset numid=3 1
5. In the terminal, enter the following command:   
    sudo cp home/pi/Desktop/LEMASdist/lightdm.conf etc/lightdm/lightdm.conf   
   This copies a file which will deactivate the blank screen screensaver so that graphed results can always be viewed without mouse/keyboard action.
6. Add a lxterminal session to launch at user login by opening the autostart file. In a terminal:  
    sudo echo '@lxterminal -e sudo /usr/bin/python3.4 /home/pi/LEMASdist/LEMASrun.py' >> /home/pi/.config/lxsession/LXDE-pi/autostart  
   This will instruct the system to open a terminal and execute the logging program upon user login.

The Raspberry Pi will begin logging and sending messages once on the REN upon restart; however, the Python logging script will need configured for the lab it will be monitoring in order for the device to identify itself correctly and send messages under the desired outage conditions. See sections *Configuring the device for a lab with existing settings*, *Adding new labs and new users*, and *Adding new lab settings*, as appropriate.

**Setting file ownership and permissions**

Open a terminal (CTRL+ALT+T) and use the following commands to change file ownership (chown) and file permissions (chmod):

sudo chown -R pi:pi /home/pi/LEMASdist/

sudo chown -R pi:pi /home/pi/Desktop/

sudo chmod u=rwx,g=rx,o=rx /home/pi/Desktop/launch-LEMAS

**Decrypting the Contacts File**

Open a terminal (CTRL+ALT+T) and use the following command to begin decryption of the Contacts.py.gpg file:

gpg /home/pi/LEMASdist/Contacts.py.gpg

Enter the passphrase provided by the curator to initialize decryption. Contacts.py will be added to the LEMASdist folder.

**Device Customization**

**Configuring the device for a lab with existing settings**

If a configuration does not exist for the lab, then the following steps must be skipped and section *Adding new lab settings* followed first.

1. In the terminal, execute the following command:   
    leafpad ~/LEMASdist/LabID.py   
   This command opens the file containing lab settings in the leafpad text editor.
2. In the leafpad text editor, navigate to the line of text that reads: labID = ‘219/G032’. This line will need updated to reflect the lab that the device will be monitoring. The format of the labID variable is, inside single quotes (‘) <building number>/<lab number>. For example, if the device will be used in building 220, room B113, the line must read:  
    labID = ‘220/B113’   
   Once the change is made, press Ctrl+S to save the change, then Ctrl+Q to quit leafpad.
3. Properly reboot the RPi by performing Start > Restart, or opening a terminal (CTRL+ALT+T) and using the command sudo reboot.

The Raspberry Pi will now begin logging upon boot.

**Changing existing lab controls**

**Adding users to an existing lab**

If a configuration exists for a lab, one must only edit Contacts.py. Note that Contacts.py.gpg must first have been decrypted into Contacts.py.

1. Open Contacts.py in a text editor:   
    leafpad ~/LEMASdist/Contacts.py
2. Find the labusers dictionary that contains the desired lab. Add a comma after the last user, before the closing square bracket ( ] ) and add the name of the user in single quotes. Their name spelling and capital letter usage must match their name in the allcontacts dictionary.

**Adding new labs and assigning contacts**

If the contacts already exist, then section the *Adding New Contacts* can be ignored. Otherwise, the contacts need added and *Adding new contacts* must be followed as well. Note: Contacts.py.gpg must first have been decrypted into Contacts.py.

1. If Contacts.py is not already opened in leafpad, in a terminal, execute the following command:  
    leafpad ~/LEMASdist/Contacts.py  
   This command opens the lab monitoring program in the leafpad tex
2. To add a new lab, navigate to the line that reads:   
    #labusers['<building>/<room>'] = ['user1', 'user2']

This is the line that needs edited. Delete the symbol for comment line (#) and replace the text inside the brackets (<>), including the brackets, with the necessary information. The names entered for users must match those in the allcontacts dictionary.

**Adding new contacts**

If a lab does not exist for the contacts, follow section *Adding new labs* first. Note: Contacts.py.gpg must first have been decrypted into Contacts.py.

1. Open Contacts.py in leafpad:   
    leafpad ~/LEMASdist/Contacts.py
2. To add a new contact, navigate to the line that reads:   
    #allcontacts['user1'] = ['<number>@<carrier>', '<email>@<domain>']   
   This is the line that needs edited. Delete the symbol for comment line (#) and replace the text inside the bracks (<>), including the brackets, with the necessary information. The spelling and capital letter usage for the user must match that in the labusers dictionary.

**Adding new lab settings**

This section is intended for when a device is added to a lab for which there are no environment settings. The settings are contained in Tcontrols.py and Rhcontrols.py, set as the Tcontrols and Rhcontrols variables, respectively. Adding users to a lab, or adding a new lab, is detailed in section *Adding new labs and new users*. The new lab should be added first.

1. In the terminal, execute the following command:   
    leafpad ~/LEMASdist/Tcontrols.py   
   to edit temperature controls or:   
   leafpad ~/LEMASdist/Rhcontrols.py   
   edit humidity controls.
2. On the last line of the file, enter a new line below. Enter into the line the format for Tcontrols:   
    ['<building>/<room>'] = [<minimum temperature>, <maximum temperature>]
3. Remove any less, than, greater than symbols. For example, for a lab in building 219, G032 with minimum temperature of 19 deg. C and maximum temperature of 21 deg. C, the line will read as follows:   
    Tcontrols['219/G032'] = [19, 21]
4. On the last line of Rhcontrols.py, enter a new line below. Enter into the line the format for RHcontrols  
    RHcontrols['<building>/<room>'] = [<minimum humidity>, <maximum humidity>]
5. Remove any less, than, greater than symbols. For example, for a lab in building 219, G032 with minimum humidity of 30 %RH and maximum humidity of 60 %RH, the line will read as follows:  
    RHcontrols['219/G032'] = [30, 60]

**Setting a date and time for a test message**

This section is intended for assigning a date and time for sending a test message to all users. This assumes the intended device is set up with users and outage conditions.

1. In Raspbian, on the device to send a test message from, open a terminal. Use the following command to open the configuration file:   
    sudo leafpad /home/pi/LEMASdist/testmsgdate.py
2. Edit the TestmsgDate variable with the date and time, in 24-hour format, a test message is desired to be sent. Surround the date and time with single quotes (‘). Use the following examples to input the date and time in the correct format:   
    TestmsgDate = ‘February, 01 2118 18:30:00’   
   or  
    TestmsgDate = ‘August, 22 1986 06:30:00’   
   Both of these settings will send a test message to all users of the device at 6:30 PM or 6:30 AM, respectively, on their respective month, day, and year. The latter will not send any messages at all because the date has already passed…hopefully. If it hasn’t, then a time paradox must have occurred because the author of this document and logging program has not yet been born. It is very important to note that setting the date and time as February 1, 2118 18:30:00 may not have the intended results. Notice the absent 0 in the day of the month. *Do not* enter any other lines, there must only be the first line containing the date and time. This format *must* be followed in order for the date and time to be parsed correctly. For reference, the strftime format is: "%B %d, %Y %H:%M:%S"   
   The website: <http://strftime.org/> details the meaning of each symbol.

**Appendices**

**Appendix A: Linux Boot Script for LEMAS using crontab: networkconfig.sh**

networkconfig.sh must be copied to /home/pi/LEMASdist/networkconfig.sh and scheduled with crontab.

networkconfig.sh file contents:

#!/bin/sh

#networkconfig.sh

# Tested with Raspbian GNU/Linux 8 (Jessie) on Raspberry Pi 3 Model B

#

#///////////////////////////////////////////////////////////////////////////////

## envlauncher.sh Notes

# Authored by: Michael Braine, Physical Science Technician, NIST, Gaithersburg, MD

# PHONE: 301 975 8746

# EMAIL: michael.braine@nist.gov (use this instead of phone)

#

# Purpose: BASH script to configure Raspberry Pi network settings on boot. Needs scheduled with crontab daemon.

# @reboot sh /home/pi/LEMASdist/networkconfig.sh

#

#///////////////////////////////////////////////////////////////////////////////

## References

# -none

#

## Change log from v1.00 to v1.00

# July 26, 2017

#

# ver 1.00 - initial version

#configure network

sudo ifconfig eth0 129.6.171.181

sudo ifconfig eth0 netmask 255.255.255.224

sudo route add default gw 129.6.171.190 eth0

alias python=python3.4

**Appendix B: Common carrier gateways**

See <https://martinfitzpatrick.name/list-of-email-to-sms-gateways/> for a more complete list.

# Carrier Email to SMS Gateway

# Alltel [10-digit phone number]@message.alltel.com

# Example: 1234567890@message.alltel.com

# AT&T (formerly Cingular) [10-digit phone number]@txt.att.net

# [10-digit phone number]@mms.att.net (MMS)

# [10-digit phone number]@cingularme.com

# Example: 1234567890@txt.att.net

# Boost Mobile [10-digit phone number]@myboostmobile.com

# Example: 1234567890@myboostmobile.com

# Nextel (now Sprint Nextel) [10-digit telephone number]@messaging.nextel.com

# Example: 1234567890@messaging.nextel.com

# Sprint PCS (now Sprint Nextel) [10-digit phone number]@messaging.sprintpcs.com

# [10-digit phone number]@pm.sprint.com (MMS)

# Example: 1234567890@messaging.sprintpcs.com

# T-Mobile [10-digit phone number]@tmomail.net

# Example: 1234567890@tmomail.net

# US Cellular [10-digit phone number]@email.uscc.net (SMS)

# [10-digit phone number]@mms.uscc.net (MMS)

# Example: 1234567890@email.uscc.net

# Verizon [10-digit phone number]@vtext.com

# [10-digit phone number]@vzwpix.com (MMS)

# Example: 1234567890@vtext.com

# Virgin Mobile USA [10-digit phone number]@vmobl.com

# Example: [1234567890@vmobl.com](mailto:1234567890@vmobl.com)

**Appendix C: Raspberry Pi configuration to use Adafruit 5” 800x480 HDMI Backpack display: config.txt**

config.txt must be copied to /boot/config.txt and overwrite the existing file.

config.txt file contents:

# uncomment if you get no picture on HDMI for a default "safe" mode

#hdmi\_safe=1

# uncomment this if your display has a black border of unused pixels visible

# and your display can output without overscan

#disable\_overscan=1

# uncomment the following to adjust overscan. Use positive numbers if console

# goes off screen, and negative if there is too much border

#overscan\_left=16

#overscan\_right=16

#overscan\_top=16

#overscan\_bottom=16

# uncomment to force a console size. By default it will be display's size minus

# overscan.

#framebuffer\_width=1280

#framebuffer\_height=720

# uncomment if hdmi display is not detected and composite is being output

hdmi\_force\_hotplug=1

# uncomment to force a specific HDMI mode (here we are forcing 800x480!)

hdmi\_group=2

hdmi\_mode=1

hdmi\_mode=87

hdmi\_cvt=800 480 60 6 0 0 0

max\_usb\_current=1

# uncomment to force a HDMI mode rather than DVI. This can make audio work in

# DMT (computer monitor) modes

#hdmi\_drive=2

# uncomment to increase signal to HDMI, if you have interference, blanking, or

# no display

#config\_hdmi\_boost=4

# uncomment for composite PAL

#sdtv\_mode=2

#uncomment to overclock the arm. 700 MHz is the default.

#arm\_freq=800

# for more options see <http://elinux.org/RPi_config.txt>

**Appendix D: Automated installation BASH script: install.sh**

#!/bin/sh

#install.sh

# Tested with Raspbian GNU/Linux 8 (Jessie) on Raspberry Pi 3 Model B

#

#///////////////////////////////////////////////////////////////////////////////

## install.sh Notes

# Authored by: Michael Braine, Physical Science Technician, NIST, Gaithersburg, MD

# PHONE: 301 975 8746

# EMAIL: michael.braine@nist.gov (use this instead of phone)

#

# Purpose: BASH script to copy and create files/folders for LEMASrun.py

#

#///////////////////////////////////////////////////////////////////////////////

## References

#

#///////////////////////////////////////////////////////////////////////////////

## Change log from v1.02 to v1.03

# November 26, 2017

#

# ver 1.03 - added decryption for Contacts.py.gpg

#

##///////////////////////////////////////////////////////////////////////////////

#ask for labID

read -p "What will be the labID? e.g. 219/G032: " labID

labIDf=$(echo $labID | sed -n -e 's/\//\_/p') #replaced / with \_ to prevent confusing filenames with directory path

#update OS

echo " "

echo "Updating Raspberry Pi"

sudo apt-get update -y

sudo apt-get upgrade -y

echo " "

echo "Removing nonessential Raspbian programs"

sudo apt-get remove minecraft-pi -y

sudo apt-get remove bluej -y

sudo apt-get remove geany -y

sudo apt-get remove greenfoot -y

sudo apt-get remove nodered -y

sudo apt-get remove scratch -y

sudo apt-get remove scratch2 -y

sudo apt-get remove sense-hat -y

sudo apt-get remove sonic-pi -y

sudo apt-get autoremove -y

echo "Raspberry Pi update done"

#install python3 libraries

echo " "

echo "Updating/installing Python3 libraries"

sudo apt-get install python3-serial #install or update pyserial

sudo apt-get install python3-numpy #install or update numpy

sudo apt-get install python3-matplotlib #install or update matplotlib

sudo python3 /usr/bin/pip install minimalmodbus #install or update minimalmodbus

#install LEMAS files

echo " "

echo "Copying LEMAS files"

CWD=$(pwd)

sudo mkdir /home/pi/LEMASdist #create LEMAS directories

sudo mkdir /home/pi/LEMASdist/tmpimg

sudo cp $CWD/LEMASRun.py /home/pi/LEMASdist/ #copy program files to LEMAS directory

sudo cp $CWD/testmsgdate.py /home/pi/LEMASdist/

sudo cp $CWD/LabSettings.py /home/pi/LEMASdist/

sudo cp $CWD/Contacts.py.gpg /home/pi/LEMASdist/

sudo cp $CWD/networkconfigs/$labIDf'\_'networkconfig.sh /home/pi/LEMASdist/networkconfig.sh

sudo cp $CWD/NoContact.list /home/pi/LEMASdist/

sudo cp $CWD/launch-LEMAS /home/pi/Desktop/

sudo cp $CWD/LEMASpi/config.txt /boot/ #move system files to operate with adafruit display and disable screensaver

sudo cp $CWD/LEMASpi/lightdm.conf /etc/lightdm

sudo amixer cset numid=3 1 #disable audio output through HDMI

#copy manuals for Comet System T3311

echo " "

echo "Copying instrument manuals"

sudo cp $CWD/LEMASmanuals/NIST-LEMAS.pdf /home/pi/Desktop/

sudo cp $CWD/LEMASmanuals/T3311manual.pdf /home/pi/Desktop/

sudo cp $CWD/LEMASmanuals/T3311specs.pdf /home/pi/Desktop/

sudo cp $CWD/LEMASmanuals/T3311protocols.pdf /home/pi/Desktop/

#copy pdf for LEMAS

sudo cp $CWD/NIST-LEMAS.pdf /home/pi/Desktop/

#change permissions and ownerships

echo " "

echo "Setting up LEMAS permissions"

sudo chmod u=rwx,g=rx,o=rx /home/pi/LEMASdist/networkconfig.sh #change permissions of bash script to be executable

sudo chown pi:pi /home/pi/LEMASdist/testmsgdate.py

sudo chown pi:pi /home/pi/LEMASdist/LabSettings.py

sudo chown pi:pi /home/pi/LEMASdist/NoContact.list

sudo chown pi:pi /home/pi/Desktop/launch-LEMAS

sudo chmod u=rwx,g=rx,o=rx /home/pi/Desktop/launch-LEMAS

sudo chown pi:pi /home/pi/Desktop/NIST-LEMAS.pdf

sudo chown pi:pi /home/pi/Desktop/T3311manual.pdf

sudo chown pi:pi /home/pi/Desktop/T3311specs.pdf

sudo chown pi:pi /home/pi/Desktop/T3311protocols.pdf

echo " "

echo "Passphrase required to decrypt contact information"

gpg /home/pi/LEMASdist/Contacts.py.gpg

rm /home/pi/LEMASdist/Contacts.py.gpg

echo " "

echo "LEMAS installation complete"

echo "Remember: LXTerminal and Chrontab need manually set up (see NIST-LEMAS.pdf in /home/pi/Desktop)"

echo "Have fun!"

**Appendix D: Configuration settings for disabling screensaver: lighdm.conf**

lightdm.conf must be copied to /etc/lightdm/lightdm.conf and overwite the existing file.

lightdm.conf file contents:

#

# General configuration

#

# start-default-seat = True to always start one seat if none are defined in the configuration

# greeter-user = User to run greeter as

# minimum-display-number = Minimum display number to use for X servers

# minimum-vt = First VT to run displays on

# lock-memory = True to prevent memory from being paged to disk

# user-authority-in-system-dir = True if session authority should be in the system location

# guest-account-script = Script to be run to setup guest account

# logind-load-seats = True to automatically set up multi-seat configuration from logind

# logind-check-graphical = True to on start seats that are marked as graphical by logind

# log-directory = Directory to log information to

# run-directory = Directory to put running state in

# cache-directory = Directory to cache to

# sessions-directory = Directory to find sessions

# remote-sessions-directory = Directory to find remote sessions

# greeters-directory = Directory to find greeters

#

[LightDM]

#start-default-seat=true

#greeter-user=lightdm

#minimum-display-number=0

#minimum-vt=7

#lock-memory=true

#user-authority-in-system-dir=false

#guest-account-script=guest-account

#logind-load-seats=false

#logind-check-graphical=false

#log-directory=/var/log/lightdm

#run-directory=/var/run/lightdm

#cache-directory=/var/cache/lightdm

#sessions-directory=/usr/share/lightdm/sessions:/usr/share/xsessions

#remote-sessions-directory=/usr/share/lightdm/remote-sessions

#greeters-directory=/usr/share/lightdm/greeters:/usr/share/xgreeters

#

# Seat defaults

#

# type = Seat type (xlocal, xremote)

# xdg-seat = Seat name to set pam\_systemd XDG\_SEAT variable and name to pass to X server

# pam-service = PAM service to use for login

# pam-autologin-service = PAM service to use for autologin

# pam-greeter-service = PAM service to use for greeters

# xserver-command = X server command to run (can also contain arguments e.g. X -special-option)

# xserver-layout = Layout to pass to X server

# xserver-config = Config file to pass to X server

# xserver-allow-tcp = True if TCP/IP connections are allowed to this X server

# xserver-share = True if the X server is shared for both greeter and session

# xserver-hostname = Hostname of X server (only for type=xremote)

# xserver-display-number = Display number of X server (only for type=xremote)

# xdmcp-manager = XDMCP manager to connect to (implies xserver-allow-tcp=true)

# xdmcp-port = XDMCP UDP/IP port to communicate on

# xdmcp-key = Authentication key to use for XDM-AUTHENTICATION-1 (stored in keys.conf)

# unity-compositor-command = Unity compositor command to run (can also contain arguments e.g. unity-system-compositor -special-option)

# unity-compositor-timeout = Number of seconds to wait for compositor to start

# greeter-session = Session to load for greeter

# greeter-hide-users = True to hide the user list

# greeter-allow-guest = True if the greeter should show a guest login option

# greeter-show-manual-login = True if the greeter should offer a manual login option

# greeter-show-remote-login = True if the greeter should offer a remote login option

# user-session = Session to load for users

# allow-user-switching = True if allowed to switch users

# allow-guest = True if guest login is allowed

# guest-session = Session to load for guests (overrides user-session)

# session-wrapper = Wrapper script to run session with

# greeter-wrapper = Wrapper script to run greeter with

# guest-wrapper = Wrapper script to run guest sessions with

# display-setup-script = Script to run when starting a greeter session (runs as root)

# display-stopped-script = Script to run after stopping the display server (runs as root)

# greeter-setup-script = Script to run when starting a greeter (runs as root)

# session-setup-script = Script to run when starting a user session (runs as root)

# session-cleanup-script = Script to run when quitting a user session (runs as root)

# autologin-guest = True to log in as guest by default

# autologin-user = User to log in with by default (overrides autologin-guest)

# autologin-user-timeout = Number of seconds to wait before loading default user

# autologin-session = Session to load for automatic login (overrides user-session)

# autologin-in-background = True if autologin session should not be immediately activated

# exit-on-failure = True if the daemon should exit if this seat fails

#

[SeatDefaults]

xserver-command=X -s 0 -dpms

#type=xlocal

#xdg-seat=seat0

#pam-service=lightdm

#pam-autologin-service=lightdm-autologin

#pam-greeter-service=lightdm-greeter

#xserver-command=X

#xserver-layout=

#xserver-config=

#xserver-allow-tcp=false

#xserver-share=true

#xserver-hostname=

#xserver-display-number=

#xdmcp-manager=

#xdmcp-port=177

#xdmcp-key=

#unity-compositor-command=unity-system-compositor

#unity-compositor-timeout=60

#greeter-session=example-gtk-gnome

greeter-hide-users=false

#greeter-allow-guest=true

#greeter-show-manual-login=false

#greeter-show-remote-login=true

#user-session=default

#allow-user-switching=true

#allow-guest=true

#guest-session=

#session-wrapper=lightdm-session

#greeter-wrapper=

#guest-wrapper=

#display-setup-script=

#display-stopped-script=

#greeter-setup-script=

#session-setup-script=

#session-cleanup-script=

#autologin-guest=false

autologin-user=pi

#autologin-user-timeout=0

#autologin-in-background=false

#autologin-session=UNIMPLEMENTED

#exit-on-failure=false

#

# Seat configuration

#

# Each seat must start with "Seat:".

# Uses settings from [SeatDefaults], any of these can be overriden by setting them in this section.

#

#[Seat:0]

#

# XDMCP Server configuration

#

# enabled = True if XDMCP connections should be allowed

# port = UDP/IP port to listen for connections on

# key = Authentication key to use for XDM-AUTHENTICATION-1 or blank to not use authentication (stored in keys.conf)

#

# The authentication key is a 56 bit DES key specified in hex as 0xnnnnnnnnnnnnnn. Alternatively

# it can be a word and the first 7 characters are used as the key.

#

[XDMCPServer]

#enabled=false

#port=177

#key=

#

# VNC Server configuration

#

# enabled = True if VNC connections should be allowed

# command = Command to run Xvnc server with

# port = TCP/IP port to listen for connections on

# width = Width of display to use

# height = Height of display to use

# depth = Color depth of display to use

#

[VNCServer]

#enabled=false

#command=Xvnc

#port=5900

#width=1024

#height=768

#depth=8