Ubuntu 20.04

**Account name**: uclp

**Pass**: security

**Root** **Password**: toor

VirtualBox tools has been installed

Software Installed:

* Git
* Vim
* curl
* Sublime-text
* pymavlink and mavproxy (sudo pip3 install --upgrade pymavlink mavproxy)
  + This gives the log analysis tools (pyulog,ulog2csv) for mavlink logs from QGroundControl
  + <https://github.com/PX4/pyulog>
  + https://www.ardusub.com/reference/data-logging.html

1. Set up PX4 Autopilot environment for SILT and HILT in folder **Documents**

* Download the source package from <https://github.com/PX4/PX4-Autopilot>
  + git clone <https://github.com/PX4/PX4-Autopilot.git>
* Run the **ubuntu.sh** with no arguments (in a bash shell) to install everything:
* bash ./Tools/setup/ubuntu.sh

1. Install QGroundControl (<https://docs.qgroundcontrol.com/master/en/getting_started/download_and_install.html>)

* Before installing QGroundControl for the first time:
  + On the command prompt enter:
    - sudo usermod -a -G dialout $USER
    - sudo apt-get remove modemmanager -y
    - sudo apt install gstreamer1.0-plugins-bad gstreamer1.0-libav gstreamer1.0-gl -y
  + Logout and login again to enable the change to user permissions.
  + Download QGroundControl.AppImage for Linux
  + Install (and run) using the terminal commands:
    - chmod +x ./QGroundControl.AppImage
    - ./QGroundControl.AppImage (or double click)

1. Run SILT with gazebo (https://docs.px4.io/master/en/simulation/gazebo.html)

Graphical user interface

Description automatically generated with medium confidence

The supported vehicles and make commands are listed below (click links to see vehicle images).

Text

Description automatically generated with low confidence

1. To run the SILT, start QGroundControl, then start the gazebo-based SILT with the drone model you select.

Load Gazebo Worlds(<http://docs.px4.io/master/en/simulation/gazebo.html#loading-a-specific-world>) (https://docs.px4.io/master/en/simulation/gazebo\_worlds.html)

SILT for mult-vehicle can be enabled by JMAVSim (<https://docs.px4.io/master/en/simulation/multi_vehicle_jmavsim.html>)

Multi-Vehicle Simulation with Gazebo (<https://docs.px4.io/master/en/simulation/multi_vehicle_simulation_gazebo.html>)

Simulating GPS Noise (http://docs.px4.io/master/en/simulation/gazebo.html#simulating-gps-noise)

What to develop:

How to simulate different types of attacks:

* Launch GPS Spoofing
  + By modifying the gazebo\_gps\_plugin.cpp in Tools/sitl\_gazebo/src, line 304- 306
    - gps\_msg.set\_latitude\_deg(latlon.first \* 180.0 / M\_PI);
    - gps\_msg.set\_longitude\_deg(latlon.second \* 180.0 / M\_PI);
    - gps\_msg.set\_altitude(pos\_W\_I.Z() + alt\_home\_ - noise\_gps\_pos\_.Z() + gps\_bias\_.Z());
  + Below are images before and after the GPS Spoofing attack
  + A picture containing ground

    Description automatically generated

A picture containing text, stationary, envelope

Description automatically generated

* Launch GPS Jamming
  + For line 36, set a counter for GPS jamming.
  + From line 334 to 356, if the counter reach 0, GPS signal lost for the UAV.

Map

Description automatically generatedMap

Description automatically generated

Text

Description automatically generated

Map

Description automatically generated

Map

Description automatically generated

* Control Command Spoofing
  + To create attacks, we need to modify code in /Users/jiawei/Documents/px4/PX4-Autopilot/src/modules/mc\_pos\_control/MulticopterPositionControl.cpp
  + This is for multicopter position control.
  + Control positions are set in method (line 196) MulticopterPositionControl::set\_vehicle\_states(const float &vel\_sp\_z)
* ADS-B Spoofing and Jamming.
* MAVLink Security

Interface Development Idea:

* Develop a command line interface that list different types of attacks
* When a particular type of attacks are selected, the program will modify the code to simulate the attack.
* Provide a second-level of manuals for the same level of attacks.
* Provide the option to restore the documents.