# Initial Setup

1. Run ‘pip install -r requirements’
2. Ensure tcpdump is installed on the system
3. Some tests require the use of selenium so a system with a GUI is required. For example, default ubuntu server install cannot be used.
4. Install the Webdriver
   1. To use selenium, install the web drive for the browser you will be using
   2. Chrome webdriver can be downloaded [here.](https://chromedriver.chromium.org/downloads)
   3. Firefox’s webdriver can be downloader [here.](https://github.com/mozilla/geckodriver/releases/tag/v0.24.0)
   4. Extract the webdrivers to a location of your choice

## File Transfer:

### Google Drive

1. Enable the Google API
   1. Go to the following [link](https://developers.google.com/drive/api/v3/quickstart/python)
   2. Select the “Enable the Drive API”
   3. If you are not signed into a Google Account, sign in or create an account
      1. This is the account used for transferring files
   4. Follow the instructions to create the app
   5. Download the credentials file and save it in the file-transfer/google-drive directory
2. Set the interface to monitor
   1. Set the variable ‘INTERFACE’ to the name of the interface to be monitored by tcpdump. The variable can be found at 17 for files ‘drive\_down.py’ and ‘drive\_up.py’
3. Add files to the directory file-transfer/google-drive/inputs to be used for uploads
4. Ensure files exist in Google Drive for downloads
5. Running the script
   1. Enter the directory file-transfer/google-drive/
   2. Before running the script, run ‘sudo -v’
   3. First time running
      1. Run 'python main.py --noauth\_local\_webserver"
      2. Visit the link displayed and select a google account to use
      3. Copy the access code
   4. Subsequent runs
      1. Run ‘python main.py’

### Dropbox

1. Enable the Dropbox API
   1. Create an account or use an existing account to sign into Dropbox
   2. Go to the following [link](https://www.dropbox.com/developers/apps)
   3. Select the "Create app" button and follow the instructions to make an app.
   4. In the app settings under the OAuth 2 section, select the "Generate" button under "Generate access token"
   5. Copy this token into the source files "dbdownload.py" and "dropboxupload.py" both at line 5 where it says "REPLACE WITH ACCESS TOKEN"
   6. Go to your account home and go into the directory "Apps > < App Name>" and create the inputs directory.
2. Set the interface to monitor
   1. Set the variable ‘INTERFACE’ to the name of the interface to be monitored by tcpdump. The variable can be found at line 4 for files "dbdownload.py" and "dropboxupload.py"
3. Add files to the directory file-transfer/dropbox/inputs to be used for uploads
4. Ensure files exist in the Dropbox Apps’s inputs directory
5. Running the Scripts
   1. Enter the directory file-transfer/dropbox/
   2. Before running any of the scripts, run ‘sudo -v’
   3. For upload, run ‘python dropboxupload.py’
   4. For download, run ‘python dbdownload.py’

## Audio Chat (VoIP):

### Skype

1. Create two skype accounts: One account for making the calls and one for receiving the calls
2. Install and login to the skype desktop application with the account receiving the calls
   1. Does not have to be on the machine running the scripts
3. Click the dial pad at the top left corner beside the search bar
4. Select the settings gear icon at the top right and go to Advanced > Answer incoming calls automatically
5. Configure the scripts
   1. In audio-chat/skype/skype.py, there are a list of variables to set

- DRIVER\_PATH ==> absolute path to the webdriver donwloaded earlier

- EMAIL ==> email address for the Skype account that will perform the calls

- PASSWORD ==> password for the Skype account that will perform the calls

- CONTACT\_NAME ==> contact name that is displayed in the contacts list of the call receiver

- CALL\_DURATION ==> duration of the Skype calls

- PCAP\_DIR ==> directory holding all the pcaps from capturing the VoIP calls. The PCAPs are named skype-voip-<Number>.pcap. If the script is run a second time, it will overwrite the pcaps in the PCAP\_DIR

* 1. In audio-chat/skype/skype.py, set the values in the for loop at line 109 to set how many calls will be created and captured

1. Run the scripts
   1. Enter the directory audio-chat/skype
   2. Run ‘sudo -v’
   3. Run ‘python skype.py’

### Facebook Messenger

An iPhone is required for automation due to its ability to auto answer phone calls. Facebook does not provide a setting or API for answering video calls.

1. Create two facebook accounts: 1 for making the call and one for receiving it.
2. Setup the receiving account on the iPhone and enable the feature for auto-answering calls
3. Set the variables at line 13 of audio-chat/messenger/messenger.py
   1. CALL\_RECEIVER is the account of the call recipient and is found in the URL
      1. Login to messenger in the browser and select the call recipient. The value is the value in the last path of the path. The URL will be in the form messenger.com/t/<friend id> and CALL\_RECEIVER will be equal to <friend id>
4. Running the Scripts
   1. Run ‘sudo -v’
   2. Run 'python messenger.py'

## Text Chat

### Telegram

1. Install the following packages
   1. pip install python-telegram-bot
   2. If an issue is encountered for installed in the cryptography package, install the following packages
      1. sudo apt-get install build-essential libssl-dev libffi-dev python3-dev
      2. If not using Ubuntu, visit: https://cryptography.io/en/latest/installation/#building-cryptography-on-linux
   3. snap install telegram-cli
      1. If you do not have snap, built telegram-cli from source: <https://github.com/vysheng/tg>
2. Create a Telegram account through download the app on either Android of iPhone
3. Create a bot
   1. Through the browser
      1. Log into the telegram in the browser <https://web.telegram.org/#/login>
   2. Through the App
      1. Start a conversation with @BotFather with the following link: <https://web.telegram.org/#/im?p=@BotFather>
   3. Type '/newbot' and follow the instructions
4. Copy the API token and set it as the value for the variable BOT\_TOKEN at line 4 in text-chat/telegram/telegram-bot.py
5. Set BOT\_NAME at line 6 in text-chat/telegram/telegram-bot.py to the username assigned to the bot
   1. The username should be in the form of @<Entered Name>Bot, where <Entered Name> is the name you gave your bot
6. Set INTERFACE at line 7 in text-chat/telegram/telegram-bot.py to the interface that will be monitored
7. The loop at line 17 sets how many messages will be sent to the bot in the PCAP. Feel free to adjust that value
8. Run telegram-cli and login with the Telegram account created
9. Run the scripts
   1. Enter the directory text-chat/telegram
   2. Telegram Bot:
      1. Run ‘python telegram-bot.py’
      2. Using the telegram app, start a conversation with bot to confirm it is running
      3. Any message sent to the bot should be sent back to you, but all letters will be capitalized
   3. Sender
      1. Run ‘sudo -v’
      2. Run 'python telegram-sender.py'

### Facebook Messenger

1. Follow the instructions at the link <https://developers.facebook.com/docs/messenger-platform/getting-started/quick-start> to set up a bot
2. Set the variables at lines 14 through 18 in the file text-chat/facebook/messenger.py
   1. CHAT\_PATH is the URL for the conversation
3. Run the scripts
   1. Run ‘sudo -v’
   2. Run 'python messenger.py'

## Email

### Yahoo

1. Create a yahoo email account for sending the emails and a second for receiving the emails
2. Set the variables at line 11 in mail/yahoo-mail/yahoo-mail.py
3. Running the Scripts
   1. Run 'sudo -v'
   2. Run 'python yahoo-mail.py'

### Gmail

1. Enable the Google API
   1. Follow the link <https://developers.google.com/gmail/api/quickstart/python>
   2. Select the "Enable the Gmail API" button
   3. If you are not signed into a Google Account, sign in or create an account
      1. This is the account that will be use for sending emails
   4. Follow the instructions to create the app
   5. Download the credentials file and save it in the mail/gmail/ directory with the python scripts for sending gmail emails
2. Running the scripts
   1. In the file spam\_mail.csv, for each receiving email address, enter it on its own line
   2. First time run
      1. Run 'python main.py --noauth\_local\_webserver"
      2. Visit the link displayed and select a google account to use
      3. Copy the access code
   3. Subsequent runs, run ‘python main.py’