**Cracker – Pcycys Project**

**Specification**

Project location: <https://github.com/sahargavriely/minions>

The final project includes a **client**, which streams md5 hashed phone number and configuration file to a **master** **server**. The server reads the configuration file and setup the **minion servers** accordingly. After that, the master sever sends the hashed password to the minions. Finely the minions crack the hashed password.

Every 10 seconds, the master server stores a txt file, which acts as a chace to save the last state. So, when the master server crushes all we need to do is to set him up again send a restart order from the client and he will continue from the last place he stopped.

**Please note:**

1. I didn’t succeed in setting up a remote server. Although, I was able to connect to remote machine copy the script of the minion server to the machine. But, unfortunately, I couldn’t run commands remotely. Hence setup minion server remotely or reset minion server in case of a crush – If it is local it’s support everything. If it isn’t local you to lift it manually.
2. I assumed the servers has python and windows operating system.

**The project**

|  |  |  |
| --- | --- | --- |
|  |  |  |
| $ git clone <https://github.com/sahargavriely/minions.git> | |  |
|  |

activation:

|  |  |  |
| --- | --- | --- |
|  |  |  |
| $ **cd** **minions**  $ venv/Scripts/activate.bat | |  |
|  |

We should be ready to roll.

(In event of virtual environment failure, all you need to do in to install the requirements.txt file)

**The master server**

The server should be available as **minion.server** and expose the following CLI:

|  |  |  |
| --- | --- | --- |
|  |  |  |
| $ **python** -m minions.server run-server \        -h/--host '127.0.0.1'          \        -p/--port 8000 | |  |
|  |

Or simply:

|  |  |
| --- | --- |
| $ **python** minions\server\masterserver.py #  With the arguments of host '127.0.0.1’ and port 8000 |  |
|  |

**The minion servers.**

The server should be available as **minion.minionserver** and expose the following CLI:

|  |  |  |
| --- | --- | --- |
|  |  |  |
| $ **python** -m minions.minionserver run-server \        -h/--host '127.0.0.1'          \        -p/--port 8000 | |  |
|  |

**the client**

The client should be available as **minions.client** and support the following CLI:

1. Upload of a configuration of the minions server and hashed password:

|  |  |  |
| --- | --- | --- |
|  |  |  |
| $ **python** -m minions.client upload-config \       -h/--host '127.0.0.1'             \        -p/--port 8000                    \        -hp/--hashedpassword 0500000000   \        'path\to\the\config-file\db.ini' | |  |
|  |

Where db.ini file can contain multiple fields. For example:

|  |  |  |
| --- | --- | --- |
|  |  |  |
| [the name of the minion]  state = up or not  local = local or not  host = host to be  port = port to be  user = username of the machine it going to run on  password = the respective password  [if it’s local all you need is]  local = yes  state = up\down  host = 127.0.0.1  port = 5559  [if it’s already up and will stay up all you need is]  host = 168.192.2.14  port = 5558  state = up  [else, it won’t work. but on theory you need]  state = up\down  local = not  host = 168.192.28.88  port = 6666  user = kingkunta  password = Passw0rd! | |  |
|  |

1. Resetting the last command due to a crush of the master server:

|  |  |  |
| --- | --- | --- |
|  |  |  |
| $ **python** -m minions.client reset-server \       -h/--host '127.0.0.1'             \        -p/--port 8000 | |  |
|  |