A Tale of Cats and a Mouse

The project consists of: *main.py*, *cordy.py*, *listy.py*, *chase_cat.py*, *mouse.py*. The necessary parameters needed in the program are stored in: *port_number*, *ukkonodes*, and *listy_location*.

main.py: In this program, *mouse.py* is called to send the mouse to a random node by ssh remote call. *listy.py* is called to run on the same node where *main.py* runs. *cordy.py* is called control the cat to search the mouse and also run on the same node where *main.py* runs. *mouse.py main.py* and *cordy.py* runs concurrently in separate thread.

mouse.py: Listen on the port assigned in the *port_number* file and wait the cat. Answer 'WOO' if searched by a cat and 'OUCH' if attacked by a cat. After being attacked and replying the 'OUCH' message, close the socket and end the thread.

listy.py: Listen on the port assigned in *port_number* file. If received new message, write it to file *cmsg*.

chase_cat.py: Runs on the remote ukko nodes to check whether there is a mouse and attack the mouse after both cats find it. It receives two input parameters: action(S or A) and catname(Jazzy or Catty). Check if there is a mouse by trying to establish a socket connection to the mouse port assigned in *port_number*. If successfully find or attack the mouse, send
F ukkoXXX catname or *G ukkoXXX catname* message to listy.

cordy.py: Send Catty and Jazzy to different nodes without overlap by utilizing the ssh remote call to run *chase_cat.py* in separate thread. It checks the *cmsg* file every 2 seconds to see if the mouse is found or caught.

All the files are stored in:
ukko004.hpc.cs.helsinki.fi/cs/home/xgli/Distributed_System_Exercise_2016/big_exercise_2/.

To run the program:
1. login to one usable ukko node listed in *port_number* and extract all the files to the directory mentioned above.

2. change the corresponding path in the source file *cordy.py, chase_cat.py,* and *main.py* with account name and ukko node host name.

3. run *main.py*.