Key logger

**Overview:**

When the key-logger is started, it will simply sit idle until it receives a start command from the WindowsServiceTracker. Once the key-logger receives the start command it sets a hook into the beginning of the Windows keyboard input hook chain. Now that the key-logger is hooked into the beginning of the keyboard input hook chain, any keyboard inputs will intercepted by our key-logger before it is able to pass to any other application's in the keyboard input hook chain. When the key-logger receives the keyboard input it will decode the key code and print it to a text file. The text file will then be accessible to the WindowsServiceTracker, which will be able to transfer the data to the server. The WindowsServiceTracker can disable key-logging by a stop command, and the key-logger will remove its hook from the chain.

**Requirements:**

* Start automatically
* Receive commands from the Windows service
* Be hidden from user’s sight
* Record keystrokes

**Interface:**

The key-logger runs as its own application. To call functions listed in its interface, and application must implement the KeyloggerCommInterface and connect to the key-logger by a named pipe.

**Public**

**SystemTrayKeylogger**

StartKeyLogger(): Sets a hook in the keyboard input hook chain and begins logging keystrokes

StopKeyLogger(): Removes its hook from the keyboard input hook chain and stops recording keystrokes

CheckIfRunning(): Returns true. Used to check that the key logger application is running.

GetKeylogFilePath(): Returns the path of the key log file as a string.

**Architecture/structure**

The key-logger is a single file program. It runs as a application in the system tray, but does not place an icon there that would alert a user of its presence. It is intended to run on startup and be called be turned on and off as needed. The callback function (HookCallback) is the most notable function. When the hook is placed in the keyboard input hook chain, it is called with every keystroke. It is responsible for parsing the key code passed in as an argument and storing it in the log file.

**Assumptions/Dependencies**

**WindowsServiceTracker**

**Overview:**

On startup, the Windows service attempts to connect to the key logger by a named pipe. This allows the service to make function calls on the key logger that are in the interface defining the pipe. If the laptop does not have a LAN connection, the service attempts to connect to an open or known Wi-Fi connection. The service then creates a thread that tries to connect to the server via TCP/IP connection. Once it connects to the server, it sends its identifier and waits for commands. It performs commands until the service shuts down or it loses its connection to the server. When it loses its connection to server it checks it begins trying to connect again in the same fashion. Alternatively, the server can tell the service that it is not stolen and it will wait for a duration before checking in again.

**Interface:**

**Tracker**

**Public:**

Tracker(): Constructor for the service

StartKeylogger():If connected to the key-logger, turns key logging on.

StopKeyLogger():If connected to the key-logger, turns the key logging off.

GetKeylogFilePath(): If connected to the key-logger, returns the string representation of the path of the key log file.

CheckIfRunning(): Returns true if the key-logger is running and the service is connected to it via the named pipe

**Protected:**

OnStart(): Automatically called when the service is started. Initializes variables and spawns a new thread to handle the TCP connection.

OnStop(): Automatically called when the service is stopping. Closes connections and joins the thread that handles the TCP connection.

**IP**

**Public:**

getTraceRoute(string hostNameOrAddress): Performs a trace route to the address given as an argument and returns the results as a list.