Python Keylogger + Botnet Controller

Written and developed by Stone Preheim

**Purpose:**

**This document will define and list the functionality to be included within a keylogger implemented using python. The keylogger will implement and extend a custom botnet controller initially conceptualized and developed within a previous project. This keylogger should be invisible in its execution to the victim while it extracts raw keyboard input and relays it to the botmaster. This project will exploit the Discord API to silently upload a series of files containing raw victim keypresses. A socket server will be included as well for direct connection and manipulation of victim filesystems. Commands will be implemented to remotely control keylogger activation and monitoring. Cross-compatibility for Windows, MacOS, and Linux will be maintained if possible and only be abandoned if system level APIs aren’t available for the intended functionality. Upon project completion malicious executables will be generated which will allow the keylogger to be easily distributed and executed without manual dependency installation. Should time allow complex functionality will be implemented such as automatic password sniffing, reboot persistence, etc.**

**Functional Specifications:**

**KEY:**

**Complete**

**In Progress**

1. **Keylogger**
   1. **Allow silent extraction of user keypresses**
   2. **Capture all keypresses regardless of application currently in focus**
   3. **Respond to controller remote commands**
   4. **Upload files periodically to discord API while activated**
   5. **Cleanup and hide any files generated through the extraction process**
   6. **Read user input at the lowest level possible, antivirus avoidance.**
2. **Botnet Controller**
   1. **Provide a system for identifying and controlling multiple users**
   2. **Implement various commands for remote control of available keylogger instances**
   3. **Maintain persistence in the event of a thread exception**
      1. **Allow botmaster to reinitiate logging without alerting user**
   4. **Allow control from both PC and mobile**
   5. **Provide socket level connections to any IP**
      1. **Must have port-forwarding configured on server target**
3. **Executables will be packaged and generated for mobile distribution**
4. **Stretch Goals**
   1. **Automatic password sniffing (Not possible if I want to keep things lightweight)**
   2. **Reboot persistence (Was going to finish this but stopped since I’m going to have others run this for demo)**
      1. **Add program to victim startup registry upon launch**
   3. **Botnet DDOS**
   4. **Silent executable relocation/duplication**
      1. **Hide executable from user once run**
   5. **Socket level keylogging (Irrelevant compared with Discord API versatility)**
   6. **Alternate output options (Irrelevant compared with Discord API versatility)**
5. **Documentation**
   1. **Documentation will be written and maintained throughout the project**
      1. **Within code comments**
      2. **Within FSD**
   2. **Time permitting tests should be written to maintain quality assurance**

**Planned Project Timeline:**

**Deliverable 1: Project FSD, demo of controller, starter code for keylogging to console.**

**Deliverable 2: MVP reached, basic remote keylogging implemented, starter code for stretch goals.**

**Deliverable 3: Malicious executables generated; some/all stretch goals depending on available time.**