WIRELESS HARDWARE KEYLOGGER USING ARDUINO LEONARDO AND NODEMCU

Bonia Jose, Elizabeth Biju, Farsana A A, Rijo M P
Department of CSE
Mar Baselios Institute of Technology and Science,
Nellimattom, Ernakulam, Kerala, India

Abstract—Keystroke logging is the process in which it will record the keys struck on a keyboard, the person using the keyboard is unaware that their actions are being monitored. The person operating the logging program can retrieve the data. A keylogger can be either software or hardware. The problem with software keyloggers are if there are multiple operating systems running, we need to separately install keylogger in each operating system. Its a tedious task. Even if we are able to overcome these there is another issue if the operating system is protected with an anti-virus, the antivirus may detect the Keylogger program as a threat. To overcome these we introduce a Hardware Keylogger. The keylogger has an added functionality, it can not only capture the data but it also supports live tethering of captured data. The setup includes an Arduino Leonardo and a nodeMCU.

I. INTRODUCTION

Keystroke logging, is also referred as keylogging or keyboard capturing. The keys struck on the keyboard is captured, typically covertly, so that person using the keyboard is unaware that their actions are being monitored. The person operating the logged program can retrieve the data. A keylogger can be either software or hardware. The problem with software keyloggers are if there are multiple operating systems running. We need to separately install keylogger in each operating system. It’s a tedious task. Even if we are able to overcome these there is another issue if the operating system is protected with an anti-virus may detect the keylogger program as a threat.

We introduce a hardware keylogger which can not only capture the data but it also supports live tethering of captured data. The setup includes an Arduino Leonardo and a nodeMCU. We are implementing this project for the federal agencies to monitoring the activities for preventing corruption. Arduino Leonardo is a microcontroller board which is based on the ATmega32u4. The board has 20 digital input & output pins where 7 pins are used as PWM outputs and 12 as the analog inputs. It also contains a 16 MHz crystal oscillator, a micro USB connection, power jack, ICSP header, and reset button that make it possible to support the microcontroller. To get started, connect it to a computer with a USB cable or power it with a AC-to-DC adapter or battery. The ATmega32u4 has built-in USB communication which makes it unique from all preceding boards by eliminating the need for a secondary processor. It also allows the Leonardo to be connected computer as a mouse and keyboard, in addition to a virtual (CDC) serial/COM port. The arduino has other implications for the behavior of the board.

Fig1.Arduino Leonardo

II. EXISTING SYSTEM

In a software key logger separate installation of key logger in each operating system is needed for multiple operating system. The OS protected with an anti-virus may detect the keylogger program as a threat.

III. HARDWARE REQUIREMENT

Some hardware components required for our proposed system.

A. Arduino Leonardo

The Arduino Leonardo is a microcontroller board which is based on the ATmega32u4. The board has 20 digital input & output pins where 7 pins are used as PWM outputs and 12 as the analog inputs. It also contains a 16 MHz crystal oscillator, a micro USB connection, power jack, ICSP header, and reset button that make it possible to support the microcontroller. To get started, connect it to a computer with a USB cable or power it with a AC-to-DC adapter or battery. The ATmega32u4 has built-in USB communication which makes it unique from all preceding boards by eliminating the need for a secondary processor. It also allows the Leonardo to be connected computer as a mouse and keyboard, in addition to a virtual (CDC) serial/COM port. The arduino has other implications for the behavior of the board.

Fig1.Arduino Leonardo

Leonardo exists in the same unit. A hotspot is created in the node MCU and keystroke is accessed. A connection is established to the Arduino Leonardo through the nodeMCU and keystroke is accessed from a remote location. The whole system works without the knowledge of the user.
Microcontroller| ATmega328  
Operating Voltage| 5V  
Input Voltage| 7-12V  
Input Voltage (limits)| 6-20V  
Digital I/O Pins| 20  
Analog Input Pins| 12  
DC Current per I/O Pin| 40 ma  
DC Current for 3.3V Pin| 50 Ma  
Flash Memory| 32 KB (ATmega32u4) of which 0.5 KB used by bootloader  
SRAM| 2.5 KB (ATmega32u4)  
EEPROM| 1 KB (ATmega32u4)  
Clock Speed| 16 MHz  

B. NodeMCU

NodeMCU is an IoT platform with low cost source. It had a firmware which runs on the ESP8266 Wi-Fi SoC which is a self contained SoC with an integrated TCP/IP protocol stack which gives access to any wifi network and also a hardware which was based on the ESP-12 module which is also a miniature wifi module. ESP32 32-bit MCU is now added to the NodeMCU.

An open source prototyping board designs are available for NodeMCU which is an open source firmware. The term NodeMCU refers to the firmware rather than the associated development kits. The firmware and the prototyping board designs are open source.

Microcontroller| ESP8266  
Working Voltage| DC 5 V  
Working Current| 15mA  
Working Frequency| 40Hz  
Max Range| 4m  
Min Range| 2cm  
Measuring Angle| 15 degree  
Trigger Input Signal TTL pulse| 10uS  

IV. SOFTWARE REQUIREMENTS

A. Arduino IDE 1.8.8

Arduino (IDE) is an open source software that helps to easily write, code and upload it to the board. It is Windows, Mac OS X, and Linux platform dependent. The environment is written in Java and based on the processing and also other open-source software is also used. This software can be used with any of the Arduino board.

Arduino is a cross-platform IDE which works in conjunction with an Arduino controller to write, compile and upload code to the board. The software provides support to Arduino boards, including Arduino Uno, Nano, Mega, Esplora, Ethernet, Fio, Pro or Pro Mini, as well as LilyPadArduino.

The universal languages for Arduino are C and C++ and thus the software is compatible for professionals who are familiar with these two. Features such as syntax highlighting, automatic indentation and brace matching makes Arduino IDE 1.8.8 a modern alternative to other IDEs.

B. PS/2 Keyboard.h Library

The PS2 keyboard allows us to use a keyboard for user input. 4 wires are coming from the keyboard and their connections are 5V- Arduino 5V, ground-Arduino GND, clock-Arduino pin 3 and data-Arduino pin 8.

C. Keyboard.h Library

The keyboard functions enable 32u4 or Software As a Medical Device(SAMD) micro based boards in order to send keystrokes to an attached computer through their micro’s native USB port. A modifier which is a special key on a computer keyboard, modifies the normal action of another key when the two are pressed at the same time. Keyboard.press() is used for multiple key presses.
D. ESP8266WiFi.h Library

The library provides ESP8266 specific Wi-Fi routines that we are calling to connect to the nework. An ESP8266 is a low-cost WiFi module that belongs to ESP’s family which you can use it to control your electronics projects anywhere in the world. It has an inbuilt microcontroller and a 1MB flash allowing it to connect to a WiFi. The TCP/IP protocol stack allows the module to communicate with WiFi signals.

E. FS.h Library

File system wrapper is a part of the ESP8266 core for Arduino environment.

F. EEPROM.h Library

The microcontroller on the Arduino has EEPROM- memories whose values are kept when the board is turned off. This library helps enable reading and writing the bytes. A different amount of EEPROM-1024 bytes on the ATmega328p, 512 bytes on the ATmega168 and ATmega8, 4 KB (4096 bytes) on the ATmega1280 and ATmega2560 are provided for the supported microcontrollers on Arduino boards. An emulated EEPROM space of 1024 bytes is given for the Arduino and Genuino 101 boards.

V. WIRING AND CONNECTIONS

Circuit for this Wireless hardware keylogger using Arduino Leonardo and NodeMCU is very simple. The circuit has an Arduino, USB keyboard and a NodeMCU. The keyboard is connected to the Arduino. The Arduino is connected to the NodeMCU. NodeMCU acts as an asynchronous http server. To connect the Arduino to the keyboard, there are 4 pins in the USB keyboard. The black pin (GND) from the keyboard is connected to the Arduino GND. The red pin (Vin) is connected to the Arduino 5V pin. The white pin (DATA) is connected to the Arduino pin D2. The green pin (IRQ-CLOCK) is connected to the ArduinopinD3. In the connection between the Arduino and the NodeMCU, a serial communication is established. For this, TX pin of Arduino is connected to the RX pin of NodeMCU. So a one sided communication is established. Inorder to power the NodeMCU, a 3VOut of Arduinoid connected to the 3Vin of NodeMCU. Also an Arduino GND is connected to the NodeMCU GND.

VI. PROGRAMMING EXPLANATION

A. Arduino

Step 1: Start
Step2: Wait for the keystroke from keyboard using Serial.available() function

Step 3: Read the incoming serial data using Serial.read()
Identify the character using the ps2keyboard library
Step 4: Send the character to the PC using Keyboard.write()
Step 5: Simultaneously Send the character to the nodeMCU
Step 6: Repeat the above steps until the device is connected
Step 7: Stop

B. NodeMCU

Step 1: Start
Step 2: Create a wireless access point with the SSID and password. Create a text file
Step 3: Start the http server
Step 4: Wait for the serial data from arduino
Step 5: Append the serial data to the file
Step 6: Tether the file using the asynchronous Http server
Step 7: Repeat the above steps until the device is connected
Step 8: Stop

VII. CONCLUSION

The Wireless keylogger is an advanced surveillance tool. Once connected there is no further physical interaction needed. So we can carry it and place it wherever we needed. This hardware keylogger supports the live tethering of captured data. This project is useful for the federal agencies to monitoring the activities for preventing corruption. The project can be used for other similar applications.

VIII. REFERENCES

[1] Zoltan Gingl, Gergely Makan, Janos Mellor Gergely Vadai and Robert Mingesz(20)- Phonocardiography and photoplethysmography with simple Arduino setups to support interdisciplinary STEM education, 10.1109/ACCESS.2019.2926519
[2] Javier Riedemann1, Ruben Pena2, Martin Perez3, Werner JAr1, Cristian Pesce4 and Pedro Melin 3(2019)- Design and building of an automatic alternator synchronizer based on Open-Hardware Arduino platform, 10.1109/ACCESS.2019.2932294
course for engineering students using open-source platforms in wireless scenarios, 10.1109/TE.2016.2526676


[8] Yun-Wei Lin, Yi-Bing Lin, Fellow, Ming-Ta Yang, and Jun-Han Lin(2017)- ArduTalk: An Arduino Network Application Development Platform Based on IoTalk, pgno.1937-9234


[12] www.elctrosome.org