Convert a wired printer into wireless with a Raspberry Pi

<u>Project realized by:</u>

Mohamed Aziz Tousli

Saifeddine Barkia

Zied Jarraya

Maher Marwani

Hazem Barka

I) Introduction:

At our school, we have the AJP Printer a.k.a Association des Jeunes Polytechniciens Printer (French name of Association of Young Polytechnicians) and it has been so annoying to move the printer from a room to another each time someone needs to use it, let along, plugging it every time, checking the connection via USB etc..

That is why, we came in with a solution to such problem: converting the wired AJP Printer into a wireless printer with a Raspberry Pi. We are five students from Tunisia Polytechnic School, and through this report, you'll find the different steps we passed through to implement such project.

With a $25 RPi Model A, a $2 power supply, a $5 SD card and a $5 of LEDs and resistors, it is possible to turn almost any wired printer to a wireless one thanks to CUPS software, a Rapsberry Pi, and some programming skills. Let's go!

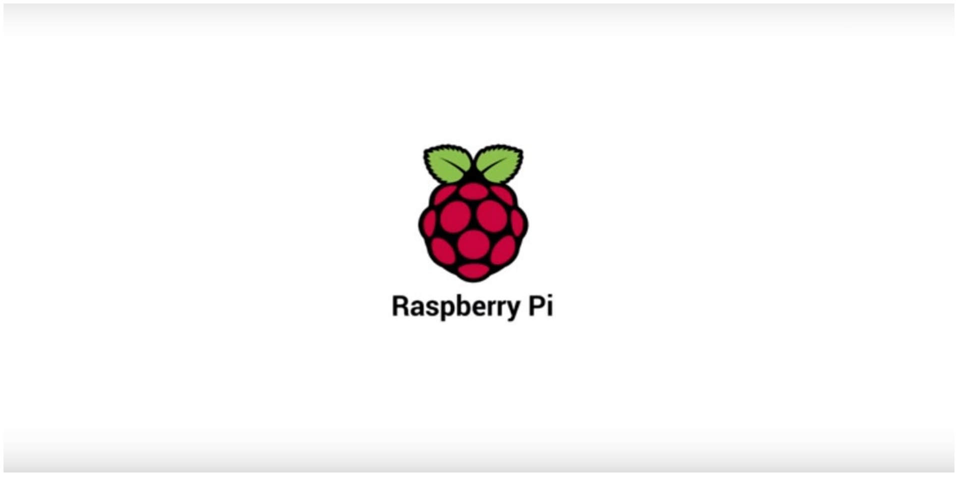
II) CUPS:

1) Set up your Raspberry Pi:

The Raspberry Pi is a single-board computer developed in the United Kingdom by the Raspberry Pi Foundation to promote teaching of basic computer science in schools and in developing countries.

Since it is a "computer", it needs an operating system, and the most common one that runs on it is called "Raspbian". Setting up the Raspberry Pi is not the primary objective of this project, that is why, you can follow this short tutorial on how to set up your Raspberry Pi:

https://youtu.be/wjWZhV1v3Pk



2) **Set up CUPS:**

As you already know, every device needs a driver, and it is the case for printers too. This is where CUPS come in. CUPS (Common UNIX Printing System) is a modular printing system for Unix-like computer operating systems which allows a computer to act as a print server. In short, CUPS is a printer driver to make Linux work more like Windows with printers.

Obviously, we will not use the “server” option. We will create our own client/server connection, using sockets, in order to print files from different computers connected to the Raspberry Pi.

1- To install it, type this into the terminal:

```
sudo apt-get install cups
```

2- After that add yourself to the CUPS admin files, with:

```
sudo usermod -a -G lpadmin username
```

3- Since CUPS doesn't have a standalone GUI, you connect to it with a web browser. You can find your IP address by typing in terminal:

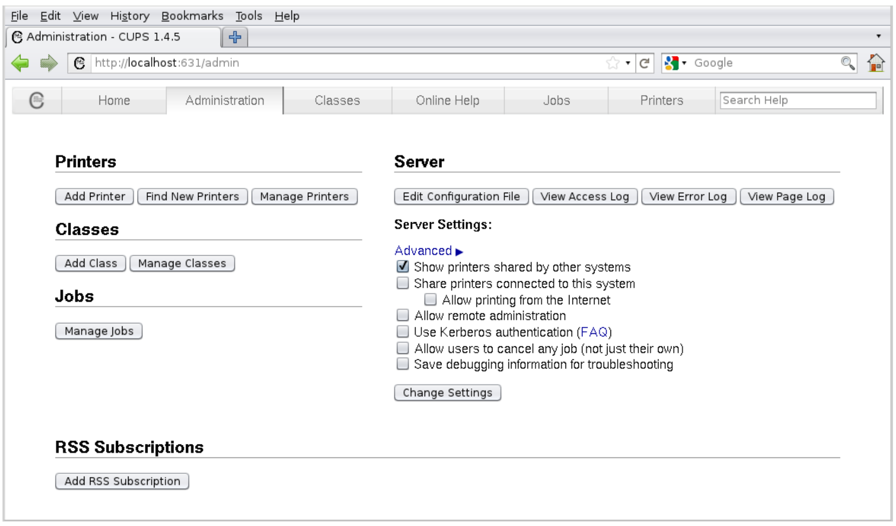
```
ip addr show
```

4- Then enter into a web browser:

```
IPAddress:631
```

PS: 631 being the CUPS default port number.

5- Click the "printers" tab and find the driver for your printer model. Install it, then print a test page.



Printers

Classes

Jobs

Server

Server Settings:

- [✓] Show printers shared by other systems
- [ ] Share printers connected to this system
  - [ ] Allow printing from the Internet
- [ ] Allow remote administration
- [ ] Use Kerberos authentication (FAQ)
- [ ] Allow users to cancel any job (not just their own)
- [ ] Save debugging information for troubleshooting

RSS Subscriptions

PS: Our AJP Printer model belongs to EPSON L220 Series.

PPS: In case of not finding your printer's driver in the drivers list, you can try to Google it, or choose the closest version to it.

III) Client/Server:

1) Server code:

Below, you can find the fully commented server code of the Raspberry Pi using sockets.

```python
import socket #Import socket library -> File recieving
import os #Import os library -> File printing through bash
import RPi.GPIO as GPIO #Import GPIO library -> Turn high/low some LEDs
import sys #Import sys library -> Exit in case of error

def errorFunction(errorString): #Function that generates the different errors

    if (errorString=="socketCreationError"):
        print("Failed to create socket!") #Print error message
        GPIO.output(redLED,True) #Turn redLED high
        GPIO.output(greenLED,False) #Turn greenLED low
        sys.exit() #Exit in case of error

    elif (errorString=="bindError"):
        print("Failed to bind socket!")
        GPIO.output(redLED,True)
        GPIO.output(greenLED,False)
        sys.exit()

    elif (errorString=="listenError"):
        print("Failed to listen to connections!")
        GPIO.output(redLED,True)
        GPIO.output(greenLED,False)
        sys.exit()

    elif (errorString=="connectionError"):
        print("Failed to accept connection!")
        GPIO.output(redLED,True)
        GPIO.output(greenLED,False)
        sys.exit()

    elif (errorString=="sendDataError"):
        print("Failed to send data!")
        GPIO.output(redLED,True)
        GPIO.output(greenLED,False)
        sys.exit()

def liveServer(s,redLED,yellowLED,greenLED): #Function for live server

    while True: #Loop for live server -> Recieve many files

        print("Server is listening...") #Print welcome message
        GPIO.output(greenLED,True) #Turn greenLED high
        GPIO.output(redLED,False) #Turn redLED low
        GPIO.output(yellowLED,False) #Turn yellowLED low

        try:
            conn, address = s.accept() #Accept request to recieve data
        except socket.error:
            errorFunction("connectionError")

        fileName = conn.recv(1024) #Recieve file name

        conn.send(b"fileName recieved") #Give the client the confirmation to send the file

        f = open('/home/pi/'+fileName,'wb') #Open new file in binary

        while True: #Loop to recieve the whole file

            GPIO.output(yellowLED,True) #Turn yellowLED high
            try:
                data = conn.recv(1024) #Recieve data
            except:
                break #Break in case of error from sender

            f.write(data) #Write data into file

            if not data: #If the buffer is empty i.e. the whole file is sent
                GPIO.output(yellowLED,False) #Turn yellowLED low
                break

        f.close() #Close the file
        conn.close() #Close the connection
        os.system("lp -d EPSON_L220_Series "+fileName) #Bash command to print the file

GPIO.setwarnings(False) #Do not show GPIO clean-up warnings

GPIO.setmode(GPIO.BOARD) #Choose broadway numbering mode

redLED = 7    #High when error is occured
yellowLED = 3 #High when someone is sending
greenLED = 5  #High when server is live

GPIO.setup(redLED,GPIO.OUT) #Set redLED to output
GPIO.setup(greenLED,GPIO.OUT) #Set greenLED to output
GPIO.setup(yellowLED,GPIO.OUT) #Set yellowLED to output

try:
    s = socket.socket() #Create a socket
except socket.error:
    errorFunction("socketCreationError")

host = "" #Accept any host
port = 8000 #Any non well-known port can be written here

try:
    s.bind((host,port)) #Create connection with host on port
except socket.error:
    errorFunction("bindError")

       try:
           s.listen(10) #Accept up to 10 connections
       except socket.error:
           errorFunction("listenError")

       liveServer(s,redLED,yellowLED,greenLED) #Call liveServer() function

       s.close() #Close the socket

```

<u>Note</u>: To make Raspberry Pi run the program in its startup, we followed this procedure:

<u>lxterminal</u> is not a good choice because it is riddled with bugs. We needed to install another terminal program, **xterm**:

```
sudo apt-get install xterm
```

Create the directory **~/.config/autostart** if it doesn't already exist:

```
mkdir -p ~/.config/autostart
```

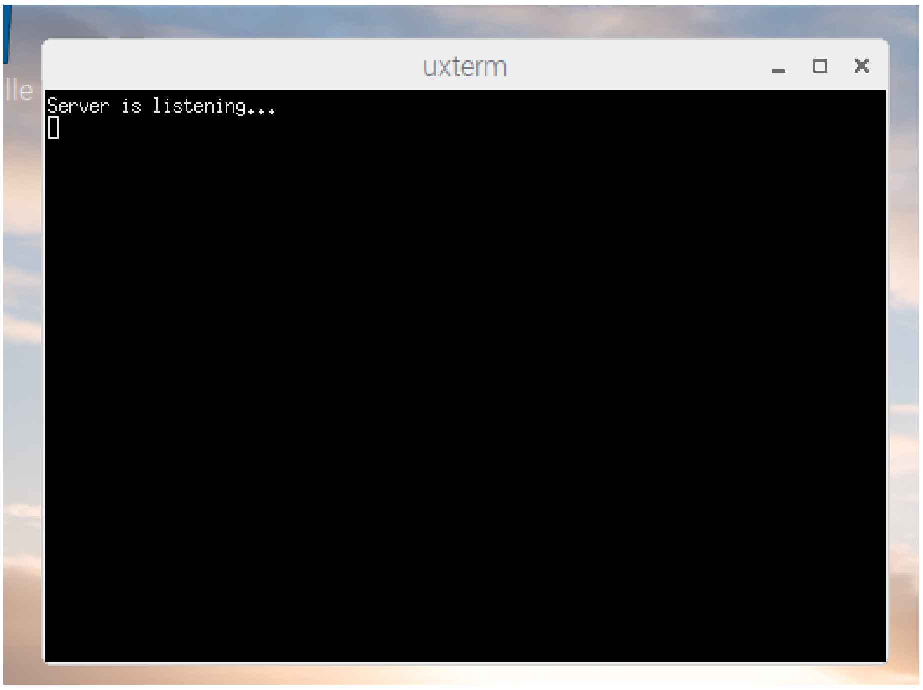
Create the file **~/.config/autostart/lxterm-autostart.desktop** with nano:

```
nano ~/.config/autostart/lxterm-autostart.desktop
```

And put this in it:

```
[Desktop Entry]

Encoding=UTF-8

Name=AJP Printer

Comment=Welcome to AJP Printer

Exec=/usr/bin/lxterm -e ‘sudo python /home/pi/ServerZ.py'
```

Now, Raspberry Pi will open this terminal window on the next startup:



```
Server is listening...
```

2) Client code:

Below, you can find the fully commented client code of the Raspberry Pi using sockets.

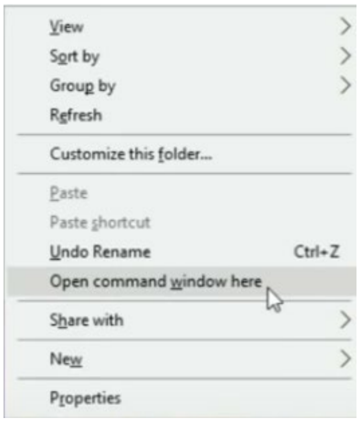
```python
import socket #Import socket library -> File sending
from tkinter import * #Import Tkinter library -> Graphic interface
import tkinter.filedialog as fdialog #-> Get fileName from the browse button
from urllib.request import urlopen #-> Get logoImage from the internet
import base64 #-> Image encoding purposes

def errorFunction(errorString): #Function that generates the different errors

    errorMessage=Label(app,text="\n"+errorString+"\n")
    errorMessage.pack()

def fileFind(): #Function to browse the file

    file = fdialog.askopenfile(filetypes=[("PDF (.pdf)","*.pdf"),("Document (.doc)","*.doc"),("DocumentX (.docx)","*.docx")])
    global fileName
    fileName = file.name

def clientCode(): #Function for client

    try:
        s = socket.socket(socket.AF_INET, socket.SOCK_STREAM) #Create a socket
    except socket.error:
        errorFunction("Failed to create socket!")

    host = "192.168.6.116" #The IP address of the server
    port = 8000 #The port used by the server

    try:
        s.connect((host,port)) #Create connection with host on port
    except socket.error:
        errorFunction("Failed to connect to host!")

    try:
        s.send(fileName.split('/')[-1].encode()) #Send fileName
    except socket.error:
        errorFunction("Failed to send fileName!")

    try:
        confirmation = s.recv(1024) #Get the confirmation from the server
    except socket.error:
        errorFunction("Server is not responding!")

    if confirmation==b'fileName recieved':

        f = open(fileName, "rb") #Open the file to be sent

        data = f.read(1024) #Send the file
        while (data):
            s.send(data)
            data = f.read(1024)

        reception=Label(app,text="\nFile sent successfully. Go take your documents from D32.\n") #Confirm sending to the user
        reception.pack()

    s.close() #Close the socket

```

```python
"""----------Tkinter Interface----------"""

app=Tk() #Create Tkinter object
app.title("Printing Service") #Make window title
app.geometry("600x400") #Make window size

#Insert TPS logo
image_url="https://i.imgur.com/ALtEouU.gif"
image_byt = urlopen(image_url).read()
image_b64 = base64.encodestring(image_byt)
photo =PhotoImage(data=image_b64)
w = Label(app, image=photo)
w.pack()
ent = Entry(app)

#Insert welcome message
welcome=Label(app,text="\nWelcome to the AJP's printing service\n")
welcome.config(font=('tahoma',15,'bold'))
welcome.pack()

#Button browse a file
browseButton = Button(app, text='Browse a File', command=fileFind)
browseButton.place(relx=0.5, rely=0.5, anchor=CENTER)
browseButton.pack()

#For organizational purposes
space=Label(app,text="\n")
space.config(font=('tahoma',2,'bold'))
space.pack()

#Button print
printButton = Button(app, text='Print', command=clientCode)
printButton.place(relx=0.5, rely=0.5, anchor=CENTER)
printButton.pack()

app.mainloop() #Let Tkinter in a loop
```

Note: In order to convert the python file clientCode.py into an executable file clientCode.exe to make it easier for the user, we followed the different steps in this tutorial:

https://youtu.be/lOIJlk_maO4

After dealing with the first part of the video, you just need to Shift + Right click next to the py file you want to convert to exe file and choose:



In the command window, you just need to type:

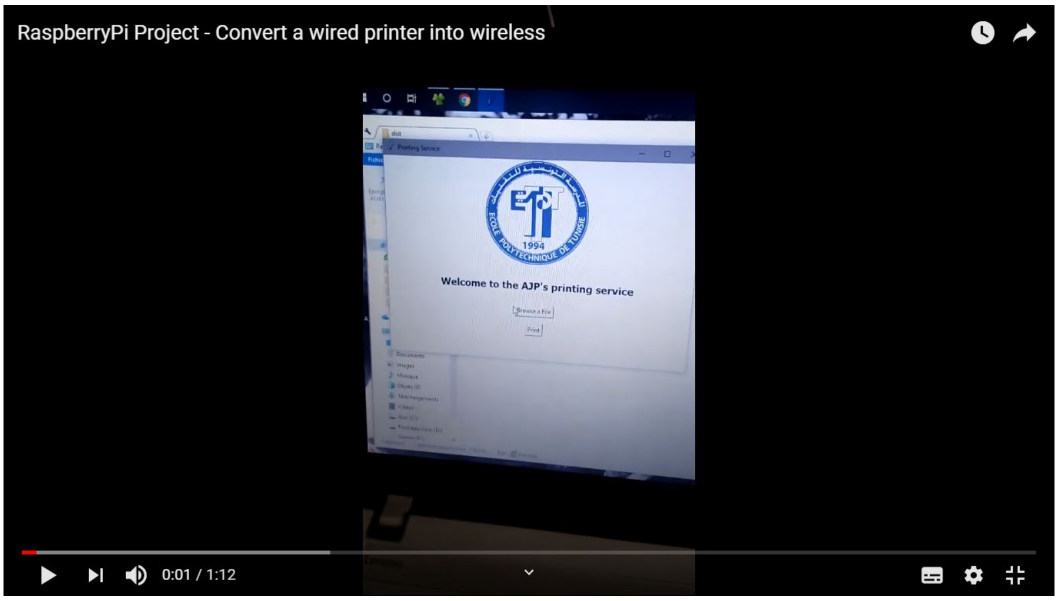
```
pyinstaller -w -F -i "C:\Users\Lenovo\Desktop\logoEPT.ico" clientCode.py
```

- -w: to remove the console that appears when you click on the executable
- -F: to make the executable in a single folder
- -i: to add an icon to the executable

3) Video simulation:

Below, you can find a video simulation of the implementation of the project:

https://youtu.be/XjJnvkOo0a0


RaspberryPi Project - Convert a wired printer into wireless
Welcome to the AJP's printing service
Browse a File
Print
0:01 / 1:12

IV) Conclusion:

To summarize, we used sockets to create a connection between the Raspberry Pi (a server) and a student from our school (a client) in order to print a file using the new AJP Wireless Printer.

We hope that you liked this project and we hope it gives you motivation to implement a similar work. As you can see, Raspberry Pi is very useful in embedded systems and Internet of Things.

V) References:

https://www.coursera.org/learn/raspberry-pi-interface

https://www.instructables.com/id/Turn-any-printer-into-a-wireless-printer-with-a-Ra/

https://www.raspberrypi.org/forums/viewtopic.php?t=66206

https://www.youtube.com/watch?v=wjWZhV1v3Pk

https://www.youtube.com/watch?v=lOIJIk_maO4