Making Operating Systems more Appetizing with the Raspberry Pi

Ziad Youssfi
Dept. of Electrical & Computer and Computer Science
Ohio Northern University
Ada, Ohio
z-youssfi@onu.edu

Abstract— **(work in progress). In 2012, Ebon Upton started the Raspberry Pi Foundation, which introduced the Raspberry Pi (RPi) as a $35 basic computer. The goal was to reignite public interest in programming and increase enrollment in college computer science. Today, the third RPi iteration features a multicore processor, a small GPU for display graphics, WiFi, Bluetooth, and USB connectivity.**

Surprisingly, even though the RPi has become very popular with the public and college students for do-it-yourself projects and maker culture, instructors have not widely used it to teach operating systems concepts. In this paper, I share my experience using the RPi in my Operating Systems class. The paper makes the following contributions:

- **Pedagogical advantages of using the RPi over other traditional and cloud-based methods to teach operating systems concepts**
- **Challenges for the instructor and students running many RPis in a classroom setting over the campus enterprise network.**
- **Advantages of using the RPi to stimulate student creativity and motivate them to design and implement their own related computing projects.**

Keywords—operating systems, Raspberry Pi, pedagogy, student motivation and learning

I. INTRODUCTION

In teaching Operating Systems, I have always chosen Linux as my OS of choice for student programming projects. Linux has many advantages:

- It is open source and free (usually under GNU license), which makes it a very attractive choice for educational institution.
- Many state of the art compilers and libraries support Linux. Current compilers and libraries cover the gamut of popular programming languages such as C, C++, Python, and Java.
- Linux provides a simple remote access with built-in secure shell (ssh), without the need for a remote desktop.
- There is a wealth of information on Linux in the literature and is referenced by many popular OS books [1].
- Current C++ standards for multithreading such the C++11, C++14, and C++17 are implemented in the latest GNU C++ compiler and library [2]. This makes Linux convenient for implementing student projects on multithreading, concurrency, scheduling, and security concepts.
- Today, a large part of the "Cloud" is run by Linux servers, as are platforms for embedded systems and mobile devices. This makes acquiring Linux skills desirable for many computer engineering and computer science students.

Using Linux in an OS class also has disadvantages:

- Since our University does not have resources to run a Linux lab or servers for students, I created a Linux server in the Cloud for student access. The remote access can be a limitation for network latency, especially for desktop access and running graphics applications.
- With a Linux server in the Cloud, students do not get the chance to manage their own OS and its security.
- Finally, the Linux shared server does not allow students to experiment with their own OS modules.

Some of these disadvantages can be addressed with virtualization (i.e. students would have to install a virtual copy on their own laptops). However, virtualization requires disk resources, a reasonably powerful x86 machine, and RAM space that students may not have on their own laptops. Also, virtualization may impose some performance penalty and introduce compatibility issues.

When I began thinking about adopting the RPi for my Operating Systems class, I was surprised not to find examples of such use for an OS class, even though the RPi has been popular on campuses for DIY projects and maker spaces. The reason for the lack of adoption might have been its low performing single core in previous versions; i.e., maybe the RPi has been viewed by many instructors as a device for hobbyists rather than for professional development or college-level courses. Instructors may also feel more at home with Microsoft Windows, which has

978-1-5090-5920-1/17/$31.00 ©2017 IEEE

dominated the market place; they may be less familiar with Linux.

II. Pedagogical Advantages of Using the Raspberry Pi

With four processor ARM cores in the newest RPi 3, students can now implement multithreading and measure speedup up to four times. Multithreading is becoming an essential part of computer engineering and computer science curricula. For example, the ACM/IEEE Computer Engineering 2016 curriculum guidelines encourage covering concurrency and multithreading in an operating system course [3]. In fact, all issues of OS topics, such as processes, threads, concurrency, deadlock, etc. can be implemented by students on the RPi 3.

Students can install and manage their own Linux. Students can get exposed to the boot-up process and can manage the OS themselves. Supported operating systems include:

- Raspian with a default desktop environment
- Ubuntu MATE with a desktop fast user interface
- Ubuntu Core: a lightweight OS suitable for embedded systems and IoT devices
- RISC OS, which is a simple OS (simpler than Linux) that might suitable for an OS class to study and modify.

Raspian as a Linux OS is not itself a real-time operating system (RTOS), but students can develop an embedded and RTOS and interface it to the RPi's GPIO. The GPIO features communication such as I2C and SPI, and serial communication. This is useful for mobile devices, IoT, robotics, and automation. The combination of multithreading on a multicore and interfacing to GPIO makes the RPi 3 a powerful embedded platform.

Moreover, students interested in gaming and multithreading can use the RPi small GPU for graphics and animation.

III. Challenges with the Campus Enterprise

Today, most computer science and electrical and computer engineering students own a laptop computer that they rely on as their primary computer. Therefore, these students must access their RPi's from their laptops over a network connection instead of having direct access. Direct access to the RPi implies connecting an HDMI display, keyboard and mouse directly to the RPi. A laptop usually cannot provide display and keyboard for an external device like the RPi.

A computer lab could provide displays for the students' RPi's, but this option limits their ability to program outside lab hours, such in the classroom or in the dorm room.

Although accessing a single RPi over the home network with a personal network router can be easily done, accessing an RPi over a secure campus network presents several challenges:

A. Finding the RPi's IP Address

In order to connect to the RPi over a network, the student has to know its IP address. On a campus network with a DHCP (Dynamic Host Configuration Protocol) service, the IP address can change, possibly for every new connection session. Of course, it is easy to get the IP address once connected, but the IP address is needed for every session before making the connection.

The Raspberry Pi organization website suggests finding the IP address by using the "nmap" (network mapper) command tool on the remote system to scan the sub-network for devices bearing the Raspberry Pi name. However, on an enterprise network, such as our campus has, DHCP clients are assigned generic names, such as vlan-101, and these names change for every connecting device. As a result, the nmap command wouldn't work on our campus and others with similar network structure.

The best way I found for my students to reliably find the IP address every time is to have the RPi email its IP address to its owner once it makes a network connection. This emailing procedure is simple and requires first installing and configuring the ssmtp tool so that RPi can use email. Configure ssmtp by editing /etc/ssmtp/ssmtp.conf with a dummy user account information that you create (for example on Gmail) as shown in Figure 1:

```
AuthUser=Dummy-User-Name@gmail.com
AuthPass=Dummy-Gmail-Password
FromLineOverride=YES
mailhub=smtp.gmail.com:587
UseSTARTTLS=YES
```

Fig. 1. Configuring ssmtp with a dummy email account

After configuring ssmtp, the "/etc/rc.local" script that runs at the end of the Linux boot-up process needs to be modified to call a script to send the email. Figure 1 shows an example of a modified rc.local file. Make sure to substitute the email that will actually receive the IP address.

```
#!/bin/sh -e
#
# rc.local
#
# This script is executed at the end of each multiuser runlevel.
# Make sure that the script will "exit 0" on success or any other
# value on error.
#
# In order to enable or disable this script just change the execution
# bits.
#
# By default this script does nothing.

# Print and email the IP address

# need to time to acquire IP from DHCP services
sleep 20
_IP=$(hostname -I) || true
if [ "$_IP" ]; then
  printf "My IP address is %s\n" "$_IP"
  /usr/local/bin/email_location.csh youremail@domain.com
fi
echo `date` " End of boot.... " >> /tmp/bootest.out

exit 0
```

Fig. 2. Modified rc.local file to call email at the end of the boot process.

The last step is to install a shell script that actually sends an email. Figure 3 shows an example of a csh script that does this task.

```
#! /bin/csh -f
set mystring = "\
                Hi, I am your RPi, and I am connected to $argv[1]\n\
                You can connect to me at the following location:\n\
                Hostname: \t`hostname`\n\
                IP:\t`hostname -I`\n\n\
                Date: `date`\n"
/bin/echo -e $mystring | mail -s "RPi Connect" $argv[2]
```

Fig. 3. script to send email with the subject "RPi Connect" that contains the IP address.

B. *Wi-Fi Certificate*

Another challenge with RPi 3 on a campus enterprise network is that the RPi needs a public certificate for its built-in Wi-Fi. Unlike home or public Wi-Fi, enterprise Wi-Fi requires a public certificate that commercial OSs can download automatically. However, for the Raspian on the RPi, the certificate must be installed manually. You will have to ask your network administrator on campus to provide specific details.

IV. Students' Motivation

When I first introduced the Raspberry Pi into my Operating Systems class, I could sense higher student motivation for programming projects than in prior semesters. Enrollment was also slightly higher with the RPi OS class. My conversation with three electrical engineering majors, who usually shy away from taking an OS class as an elective, revealed that their main reason for taking class was to learn how to program the RPi.

To quantify student motivation and excitement, I conducted a short survey at the end of the semester. Here are the survey results regarding the use of the RPi in the OS class:

Rate the use of the RPi in the OS class	***Respondents***	***Mean answer (scale 0 to 5)***
Your curiosity to use the RPi	15	4.63
Your excitement to buy the RPi 2 or RPi 3	15	4.27
Your motivation to buy extra accessories	15	3.77
RPi usefulness to apply concepts such as concurrency, deadlock, processes, and threads	15	4.40
How the RPi promoted your interest in programming and DIY technology	15	4.47

Concerning future use of the RPi after the class, I asked the students the following question:

	Respondents	***Mean answer (scale 0 to 5)***
In the future, how likely are you to make something that would involve the RPi?	15	4.67

To be more specific about hardware and software features the students might use in the future, the survey asked the following:

In the future, what RPi feature(s) would you be interested to use (pick all that apply)?	***Number of respondents who picked this option***
HDMI	11
Display interface	12
GPIOs	11
Sound	8
Camera interface	6
Sensors, such temperature, pressure, gyroscope, etc.	13
Others, please specify: gamepads	1

I asked about the types of software applications the students might develop in the future:

In the the future, what kind of software would you create with the RPi (pick all that apply)?	***Number of respondents picking this option***
3D Graphics	4
Games and entertainment	10
Home automation	14
AI and pattern recognition	8
Others, please specify:	0

The survey asked about the students' level of expertise in Linux:

	Respondents	***Mean answer (scale 0 to 5)***
What was your level of expertise in Linux before the class?	15	1.53
In the future, how useful do you think Linux expertise will be for you?	15	4.07

The above table shows that most students came into the class with low level of expertise in Linux. Yet, most of them believed that in the future Linux will be useful to them, which support the use of Linux on the RPi in the OS class.

Finally, here are some general comments the students left at the end of the survey:

I very much enjoyed the hands-on experience with using the Raspberry Pi
I learned a ton about C and Linux. This class was helpful.
I think it was great getting some hands on experience with the pi
I really enjoyed using the RPi. It was great to learn more about Linux as well as OS's.
Loved the course!

V. Conclusion & Future Directions

Although the RPi has been popular on college campuses for DIY projects and maker spaces, I could not find much adoption

of it in operating systems classes. This paper shares the experience of adopting the Raspberry Pi 2 and 3 as a platform for teaching operating system concepts. The RPi 3 boasts four core processors and its Linux operating system can be adopted to illustrate concurrency and multithreading concepts that are essential for the OS class. Having their own RPi's, students can leverage the Linux open source to experiment with developing and interfacing their own OS modules. Using the RPi GPIO and communication, students can also experiment with real-time operating systems (RTOS). Accessing the RPi over an enterprise network poses some challenges, for which I propose solutions. Finally, I show a survey from my operating system class that shows students more excited and motivated not just about the class but also about future use of the RPi.

In the future, I would like to develop more graphics programming exercises that would take advantage of the RPi GPU. I also would like to create more exercises that would leverage the RPi as an embedded platform. I think the combination of embedded system and graphics would make the RPi an even more compelling platform for computer engineering and computer science students.

REFERENCES

[1] William Stallings, "Operating Systems, Internals and Design Principles", Pearson, 8th edition, 2016.

[2] Anthony William, "C++ Concurrency in Action", Manning, 2nd edition, 2017.

[3] ACM/IEEE, *Final Curriculum Report 2016*. Available: http://www.acm.org/binaries/content/assets/education/ce2016-final-report.pdf