Security Vulnerabilities in Raspberry Pi–Analysis of the System Weaknesses

Article *in* IEEE Consumer Electronics Magazine · November 2019

DOI: 10.1109/MCE.2019.2941347

CITATION
1

READS
6,693

4 authors:

Jorge Sainz Raso
National Distance Education University
2 PUBLICATIONS **1** CITATION
SEE PROFILE

Sergio Martín
National Distance Education University
191 PUBLICATIONS **2,178** CITATIONS
SEE PROFILE

Gabriel Diaz
UNED - Spanish National Distance Education University
189 PUBLICATIONS **1,923** CITATIONS
SEE PROFILE

Manuel Castro
National Distance Education University
597 PUBLICATIONS **5,085** CITATIONS
SEE PROFILE

Security vulnerabilities in Raspberry Pi. Analysis of hardware and software weaknesses

By J. Sainz-Raso, S. Martin, G. Diaz, and M. Castro

Abstract-The Internet of Things (IoT) is made up of many devices, platforms and communication protocols. Among them, Raspberry Pi has arisen as one of the most popular equipment for hobby and education purposes because of its low cost, small size, flexibility and potential. Nevertheless, the Raspberry Pi needs an operating system to work, which exposes it to software vulnerabilities despite the many advantages it provides in comparison with non-operating system devices. This device also has hardware limitations which impact on its security. These limitations exist because some concessions had to be done during the design in order to decrease the cost of the device. This article analyzes different hardware and software vulnerabilities that can be found in a Raspberry Pi when using a default installation of different available operating systems. As a result of this study, we propose a list of good practices to minimize the presented issues.

1. INTRODUCTION

A system exposed to the Internet is always a risk to itself and its surrounds [1]. Any connected device, including cars or appliances, is vulnerable. If a single device of their environment is easily hackable, it will be easier for an attacker to gain control of more interesting devices. But vulnerabilities can also be a part of the hardware if design is not good enough or if malicious non-authorized parties have access to the hardware at any of the fabrication steps [2].

Due to its price, size, potential and flexibility, the Raspberry Pi is a perfect device for IoT environments and its use has been exploited as an IoT device. It is a small, low cost, generic- purpose computer that costs around 40€ and works the same way as a classic computer does: it requires a mouse, a keyboard, a screen and a power supply. According to [3] the Raspberry Pi sales are already over the 19 million units among all of its models (Figure 1).

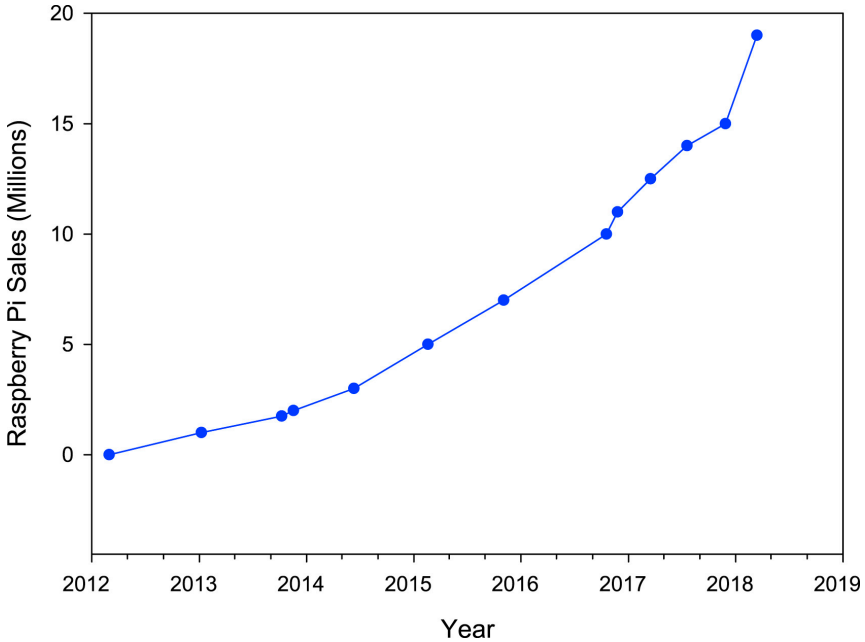

Fig. 1. Raspberry Pi units sold (all models), according to statistics published by the official Raspberry Pi blog [3].

The growing number of Raspberry Pi users implies more and more unprotected systems connected to the Internet. Currently, installing an operating system (OS) for the Raspberry Pi is very simple; and this may lead some users to leave the default configuration. This implies a high security breach, as it potentially contains vulnerable non-required services. Default installed users could give an attacker free way to access the system. The Raspberry Pi also has hardware limitations and vulnerabilities that must be known in order to avoid or minimize their effects.

Analyzing the literature about security analysis performed on Raspberry Pi, only few works have been done. Vasudevan et al [4] proposes a security architecture for improving Raspberry Pi software security but without providing an exhaustive analysis of the OS available for the device. Others, such as [5], focused on security analysis of Raspberry Pi against particular cryptography algorithms, such as RSA. None of them provide hardware or software analysis of this device to help engineers, hobbyist, designers and developers in their projects.

The goal of this article is to analyze both hardware and software vulnerabilities of Raspberry Pi system after a default installation. The conclusions of this paper provide some good practices guidelines that can be applied in order to limit the presented vulnerabilities.

II. METHODOLOGY

The stages involved in the study involved both hardware and software vulnerabilities analysis to cover all possible limitations in the device. The hardware design study included the external connectors, the processor and the internal design. For the software analysis, the most common OS for Raspberry Pi (i.e. Raspbian, Windows 10 IoT, Open- ELEC, Ubuntu and RiscOS) were studied, from the point of view of security, after a default installation (default root user and the default installed services. In case a service was vulnerable to a known bug, an attack to the service was also performed. The utilities or methods used to analyze the systems were NMAP, W3AF, Metasploit. Finally, after evaluating all presented vulnerabilities, a set of good practices is proposed.

III. HARDWARE ANALYSIS

The design of the Raspberry Pi does not fully focus on safety and it owns some vulnerabilities and limitations that **should** be known. Some limitations are due to the decisions taken on the design to keep the device cheap. This section shows some vulnerabilities or limitations the Raspberry Pi has.

A. USBs power and Backfeeding

Depending on the version of the Raspberry Pi, the output current on the USBs can go from 500mA to 1.2A [6], which is not always enough to power a device. In order to avoid this limitation, a powered USB hub can be used. A powered USB hub can also be used to backfeed the Raspberry Pi. Backfeeding means that the Raspberry Pi will not be fed using the normal power source; instead it will be powered from the incoming current from one of the USB ports. This must be done carefully because any current over 2.5A may destroy the Raspberry Pi [6] due to the lack of USB protections.

B. Overclocking

Overclocking the Raspberry Pi has always been possible, but since September 2012 Raspbian includes a safe option to perform it through the raspi-config application. If this option is used, the system will automatically disable overclocking if temperature reaches 85°C. Any other attempt to overclock the Raspberry Pi will permanently set a bit inside the SoC, which can be checked in the file /proc/cpuinfo. Revision information that starts by 1000 implies that the Raspberry Pi has been overclocked outside of the allowed limits.

C. GPIO logic levels and Serial access

The GPIO pins of the Raspberry use a voltage of 3.3V and do not have any protection against overvoltage. This means that any voltage higher than 3.3V may destroy the block inside the SoC, so connections must be done very carefully. As an example, the GPIO pins can be used to access the console

through the serial port, but a logic converter will be required as the serial port uses 5V logic.

D. Real time clock absence

In order to reduce costs, the Raspberry Pi does not have a real time clock, given that it would have increased the final cost because of the extra area requirements to hold the clock, the battery and other required components to make it work. The absence of a real time clock means that date and time are not stored internally, which is critic for functionalities such as certificate validation or other cryptographic actions. In order to recover the correct time after reboot, the Raspberry Pi can get the hour from a Network Time Server, which implies having a network connection [7].

E. Xenon flash shyness

The Raspberry Pi 2 reboots if it is pointed with a laser pointer or a xenon flash [8]. This kind of light confuses the processor core power regulator and generates a voltage drop, which makes the Raspberry Pi 2 reboot. A simple solution is to cover the chip marked as U16 with an element such as Blue-Tack.

IV. SOFTWARE ANALYSIS

The Raspberry Pi needs an OS in order to work. There are some OS ready to be downloaded and easily installed from the official Raspberry Pi website. In this section, an analysis of the security of some of the systems that can be installed on the Raspberry Pi 2 will be performed. The analyzed OS are: Raspbian, Windows 10 IoT, OpenELEC, Ubuntu and RiscOS. For each system, we have analyzed: 1) default users and passwords; 2) available public services; 3) if possible, an attack to the public services.

A default password is a security breach in a system, given that any attacker could easily access the system after discovering the system type [**9**]. Unnecessary services can also be a source of security issues as they may be misconfigured and exposed to bugs without providing any useful.

A. Raspbian

The two analyzed versions of Raspbian are Wheezy, released on May 2015; and Stretch, released on August 2017. The installation of a Raspbian system creates the default user –pi- and its default password -raspberry. This system does not let the user choose this username and password during the installation. Only after finishing the installation the user may change the password, as they would do in a regular Linux system.

Regarding the public services of Weezy, NMap only discovers an SSH server in the port 22. This is an OpenSSH server, version Debian-4+deb7u2 1.0.1e 11 Feb 2013. This version could be vulnerable to the UserRoaming bug, but it is not. This is because the stdio buffers used in GNU/Linux are cleaned after use [**10**]. Concerning the Heartbleed bug, this version is not vulnerable to it because the package installed in this version of Raspbian (OpenSSL 1.0 2+rvt+deb7u16) has the bug fixed **[11]**. Stretch does not have any public service according to NMap. This result is expected as Raspbian has its SSH server disabled by default since the release of November 2016 [**12**]. This service can be activated from the Raspbian configuration menu.

B. Windows 10 IoT

For the Windows 10 IoT analysis, two Windows IoT versions have been analyzed: Insider Preview 10.0.10556.0, from November 2015 and Insider Preview 10.0.15063.0, from September 2017. Windows 10 IoT 10.0.10556.0 has a default administrator user -Administrator- and password -p@ssw0rd. This password can be changed after the installation using the command line.

A default installation of this OS keeps eight open ports with their services. These services are: FTP (port 21), SSH (22), Microsoft Windows RPC (135), Samba (445), remote debug (4020), remote management (5985), web server (8080), AllJoyn (9955) and WinRM (47001). Among these services, the most interesting ones are FTP and HTTP. The FTP service is an anonymous FTP server with read and write rights on all folders, including system ones. This means that anyone that can access the Raspberry

Pi will have complete control of the system as they could modify -or delete- any of its configurations.

Regarding the default administrator password, it can now be chosen if the IoT Dashboard utility is used to create the image. Moreover, the FTP server does not run anymore as a start-up daemon [**13**]. In both tested versions, the configuration of Windows 10 IoT can be set up using the Windows Device Portal, available through the web service on the port 8080. The access to this service requires authentication, but it is done in clear text, so it is easy to crack. An attack against this service using man in the middle and ARP-spoofing can be performed and the password of the administrator can easily be discovered using Wireshark.

C. OpenELEC and LibreELEC

The version of OpenELEC analyzed in this article is the 8.04 and the LibreELEC one, 8.02. For both systems, only a Samba server is installed by default. Both systems are installed by default with the user –root- and the password –openelec. These parameters cannot be changed. If a user needs to change these values, he will need to recompile the whole system from its sources [14].

The two systems have two easy-to-access services that do not require any authentication to access to them: an HTTP server and a Samba server. The HTTP server provides, through a web interface, complete control of the system as if it was a remote controller. The Samba service gives access to the media -so it can be handled easily- and it also shares some configuration files of the system. This shared information includes Samba's configuration, so it would be easy to modify the configuration file to share the whole system as a resource.

All these shown security issues are already known. In their forums it is warned that OpenELEC is only an entertainment system and that it has not been designed targeting security [14]. As LibreELEC is a fork of OpenELEC, this can also be applied to it.

D. Ubuntu

The Ubuntu version used for the analysis is 16.04. By default, no services are installed, but for this exercise Open SSH has been installed during installation. Ubuntu creates a normal user with superuser rights instead of asking for the root password creation. The username and the password of this user will be asked during the installation. This avoids the creation of default users or passwords, increasing security. NMap shows, as expected, a single public service which is an SSH server listening in the port 22. The installed versions of OpenSSL (1.0.2g) and OpenSSH (7.2) are neither vulnerable to Heartbleed nor to UserRoaming bugs.

E. RiscOS

RiscOS is an OS created in 1987 by Acorn Computers. It was designed to run on RISC processors, which is precisely what ARM is. This O.S. is a single user operating system. This means that only one user can be logged in at the same time. It also counts on WIMP (Windows, Icons, Menu and Pointer) to let the user interact with the system [6]. The RiscOS version analyzed is the 15, released on May 2017. This version can be downloaded from the official website of Raspberry Pi and installed easily in an SD card. After a default installation, NMap cannot discover any public service. This means that the system cannot be accessed remotely. If someone wants to access the system, they will require physical access to the hardware, screen and input devices -keyboard and mouse. Taking into account these facts, RiscOS is a highly secured system after a default installation.

V. DISCUSSION AND CONCLUSIONS

This article has analyzed some hardware weaknesses and limitations as well as OS vulnerabilities of a default installation of Raspberry Pi. A list of hardware constraints can be seen on Table I.

After analyzing the hardware, it can be concluded that the Raspberry Pi hardware design is more focused on decreasing costs than on security. This becomes obvious when observing hardware limitations such as the absence of a real time clock, or the lack of USB protection. Weaknesses like the xenon flash

shyness or the sensitivity to 5v logic on the GPIO pins are also examples of how the hardware design is focused on costs.

Users should be aware of this design goal and must put extra effort on securing it. Physical access of non-authorized people must be avoided. If untrustworthy people have access to the physical system, they will be able to manipulate the environment easily: shutdown the system, change network connections or even connect to the system using its ports. In the case of Raspberry Pi 2, it is enough to take a picture with flash to reboot the system. Hardware must be manipulated carefully. A simple mistake when connecting something with the wrong voltage to the GPIO pins can destroy the Raspberry Pi. Bad quality powered USB hubs may also damage the Raspberry Pi if the output voltage is above 5V or if the output current is over 2.5A.

TABLE I.
SUMMARY OF HARDWARE VULNERABILITIES AND LIMITATIONS

Vulnerability	Description
USB power and backfeeding	• Powered USB hub required for USB devices that consumes more than 500 mA • Raspberry Pi can be powered by its USB due to the lack of USB protections
Overclocking	• Bad configuration can burn the device
GPIO logic levels and serial access	• GPIO uses 3.3V logic. 5V logic may destroy the SoC
Real time clock absence	• Raspberry Pi cannot keep the time after being powered off
Xenon flash shyness	• RPi 2 reboots if it is pointed with a laser or a xenon flash

Regarding the OS analysis, if the system is exposed to the Internet, setting the correct configuration is a must. Table II shows a summary of OS fragilities. Default users and passwords and unsecured services are the main issues detected. Default user vulnerability can give an attacker free way to enter the system without being noticed. In order to avoid this problem, a user should change the default root password after installation. Unsecured services can also give an attacker an easy way in to get system credentials. This is why a correct configuration of required public services must also be performed. It is also required to remove unnecessary public services after installation.

TABLE II.
SUMMARY OF OS VULNERABILITIES

Operating system	Vulnerability
Raspbian	• Default user and password
Windows 10	• Unsecure Windows Device Portal
OpenELEC LibreELEC	• Default user and password (and cannot be changed) • HTTP & Samba unsecured services
Ubuntu	• Nothing to report
RiscOS	• Nothing to report

As conclusion, a weak system in a network exposes the whole environment, so users must understand the risk of not spending time on securing a system. The Raspberry Pi is a low-cost complete computer, which implies some design and production concessions. This leads to some limitations and weaknesses that must be taken into account when working with it. A default configuration might leave the system in a vulnerable state, something that can be exploited by malicious third parties. Being aware of the vulnerabilities of this popular device will help many engineers, developers and designers to follow good practices to avoid or minimize the risk of attacks on production systems.

ABOUT THE AUTHORS

Jorge Sainz Raso (jorgesr86@gmail.com) is a computer engineer currently working in a private company in Lyon (France) as a software engineer. He obtained his degree in the University of Zaragoza. He has a Master's Degree on Communication Networks and Content Management by UNED.

Sergio Martin (smartin@ieec.uned.es) is Associate Professor at the Electrical and Computer Engineering Department of UNED. He is a Computer Engineer and he obtained his PhD in 2010 from the same University. He authored more than 150 peer-reviewed publications. He is IEEE Senior Member.

Gabriel Diaz (gdiaz@ieec.uned.es) is Associate Professor at the Electrical and Computer Engineering Department of UNED. He obtained a PhD in Physics from Universidad Autónoma de Madrid. He is author and co-author of over 150 peer-reviewed publications. He is IEEE Senior Member.

Manuel Castro (mcastro@ieec.uned.es) is Full Professor at the Electrical and Computer Engineering Department of UNED. He obtained a PhD in Industrial Engineering from Universidad Politécnica de Madrid. He is author and co-author of over 200 peer-reviewed publications. He is IEEE Fellow Member, President of the IEEE Education Society (2013-2014) and Director-Elect of IEEE Division VI (2018).

ACKNOWLEDGMENTS

The authors acknowledge the support provided by the projects: ETSII/UNED 2019-IEQ13, e-Madrid-CM (P2018/TCS-4307) and IoT4SMEs(2016-1-IT01-KA202-005561).

REFERENCES

[1] J. H. Lee and H. Kim, "Security and Privacy Challenges in the Internet of Things [Security and Privacy Matters]", in *IEEE Consumer Electronics Magazine*, vol. 6, no. 3, pp. 134-136, July 2017.

[2] A. Sengupta, "Hardware Vulnerabilities and Their Effects on CE De- vices: Design for Security Against Trojans [Hardware Matters]" in *IEEE Consumer Electronics Magazine*, vol. 6, no. 3, pp. 126-133, July 2017.

[3] S. J. Johnston, P. J. Basford, C. S. Perkins, H. Herry, F. P. Tso, D. Pezaros, R. D. Mullins, E. Yoneki, S. J. Cox, J. Singer, "Commodity single board computer clusters and their applications", *Future Generation Computer Systems*, Volume 89, 2018, Pages 201-212.

[4] A. Vasudevan and S. Chaki, "Have Your PI and Eat it Too: Practical Security on a Low-Cost Ubiquitous Computing Platform," in *2018 IEEE European Symposium on Security and Privacy* (EuroS&P), London, 2018, pp. 183-198.

[5] A. Sanada, Y. Nogami, K. Iokibe and M. A. Khandaker, "Security analysis of Raspberry Pi against Side-channel attack with RSA cryptography," in *2017 IEEE International Conference on Consumer Electronics - Taiwan (ICCE-TW)*, Taipei, 2017, pp. 287-288.

[6] Raspberry Pi FAQs. "What are the power requirements?" URL: https://www.raspberrypi.org/help/faqs/#powerReqs. Last access: 01/09/2017

[7] B. Horan, "Practical Raspberry Pi", *Technology in action*, 2013.

[8] Raspberry Pi Official Site. "Xenon death flash: a free physics lesson". URL: http://www.raspberrypi.org/xenon-death-flash-a- free-physics- lesson/. Last access: 01/09/2017

[9] US-Cert. Alert (TA13-175A): Risks of Default Passwords on the Internet.

[10] Qualys Security Advisory, "Roaming through the OpenSSH client", CVE-2016-0777 and CVE-2016-0778.

[11] Raspbian Bug Tracking, "Raspbian openssl (1.01e) is vulnerable to CVE-2014-0160".

[12] Raspberry Pi Documentation: SSH (Secure Shell). URL: https://www.raspberrypi.org/documentation/remote-access/ssh/. Last access: 03/09/2017

[13] Release Notes for Windows 10 IoT Core 15063. URL: https://developer.microsoft.com/en-us/windows/iot/docs/releasenotesrtm. Last access: 01/09/2017

[14] OpenELEC FAQ: How do I change the SSH password? URL: https://wiki.openelec.tv/index.php/OpenELEC_FAQ. Last access: 01/09/2017