Assessment and Hardening of IoT Development Boards

Omar Alfandi[2(✉)], Musaab Hasan[1], and Zayed Balbahaith[2]

[1] University of Science and Technology of Fujairah, Fujairah, United Arab Emirates
m.mohammad@ustf.ac.ae
[2] Zayed University, Abu Dhabi, United Arab Emirates
{omar.alfandi,M80007225}@zu.ac.ae

Abstract. Internet of Things (IoT) products became recently an essential part of any home in conjunction with the great advancements in internet speeds and services. The invention of IoT based devices became an easy task that could be performed through the widely available IoT development boards. Raspberry Pi is considered one of the advanced development boards that have high hardware capabilities with a reasonable price. Unfortunately, the security aspect of such products is overlooked by the developers, revealing a huge amount of threats that result in invading the privacy and the security of the users. In this research, we directed our study to SSH due to its extensive adoption by the developers. It was found that due to the nature of the Raspberry Pi and development boards, the Raspberry Pi generates predictable and weak keys which make it easy to be utilized by MiTM attack. In this paper, Man in The Middle (MiTM) attack was conducted to examine the security of different variations provided by the SSH service, and various hardening approaches were proposed to resolve the issue of SSH weak implementation and weak keys.

Keywords: IoT · Raspberry Pi 3 · Man-in-the-middle attack · Remote authentication · SSH keys · OpenSSH

1 Introduction

The great advancements in the internet and its extremely high speeds revealed a huge amount of innovative inventions that are based on the internet, releasing the Internet of Things (IOT) term and even the Internet of Everything (IoE) [1]. These products that depend on their operation mainly on the internet started to be an essential part of any home [2]. Recently, the development of internet-based products become not limited to specialized manufacturers only but became on hand to small developers with limited resources due to the widely available development boards that could be bought from the market with high hardware specifications and low cost [3]. IOT has continuously filled all aspects of contemporary human life, such as learning, healthcare, and business, involving the warehouse of sensitive data about people and companies, commercial data, product development, and marketing [4]. The cluster appropriation of the internet of things (IOT) is a multibillion-dollar chance for product companies, An

© IFIP International Federation for Information Processing 2019
Published by Springer Nature Switzerland AG 2019
M. Di Felice et al. (Eds.): WWIC 2019, LNCS 11618, pp. 27–39, 2019.
https://doi.org/10.1007/978-3-030-30523-9_3

expected 30 billion devices or things will be connected to the internet by 2020, with a cost expected to be $1.7bn [5]. In future, IOT will ultimately improve our living behaviors and allow people and devices to interact anytime, anyplace, with any device under typical circumstances using any network and any service [6].

The main purpose of IOT is to produce a better environment for humans in future [7]. IOT computation different hardware-based components and platforms, in the modern times, hardware devices identified as single board computers have been improving into affordable, powerful, and skilled machines, the most popular at the time is the Raspberry Pi, which produced by Dr. Eben Upton and the Raspberry Pi foundation in 2010 [8].

The Raspberry PI device formed to facilitate for people to explore computing and to learn programming languages like Python and Scratch [9]. Since its beginning, the device has moved on to be adopted in home projects as well being practiced by education institutes to be assistance in educating student's computer science concepts. This small size computer device can arrange everything that a normal computer can do from browsing the net, enjoying video games, forming word files, etc. [10]. Raspberry PI has a processor, memory and graphics driver for output HDMI. It has the ability of plug-in on a computer screen, keyboard and mouse. It can also be utilized in various apps requiring interacting with the outside world [11]. Raspberry PI matches all the characteristics of being an IOT device which can be configured to produce a different kind of functionalities per user conditions. This small device has been employed in various and several applications and can be incorporated into networks, which has to commence to questions concerning about security weaknesses regarding such device [9].

The widespread distribution of devices in the IOT has created tremendous demand for strong security in response to the increasing demand of billions of connected devices and services [12]. A number of threats are rising every day, and attacks have been on the rise in both number and complexity, also the tools accessible to potential attackers are also growing more complex, productive and efficient [13]. IOT challenges an amount of threats that need to be noticed for protecting actions to be taken. Unluckily, the majority of these devices and apps are not outfitted to manage the security and privacy attacks and it raises a lot of security and privacy concerns in the IOT networks [14]. An assessment reveals that 70% of the IOT devices are very easy to attack; therefore, an effective mechanism is greatly required to secure the devices connected to the internet against hackers and intruders [15]. Security requirements in the IOT context are not dissimilar from any other ICT systems; therefore, for IOT devices to reach the completest potential, it needs protection against threats and vulnerabilities [16].

In the Raspberry PI, a security issues of generation a predictable secure shell (SSH) keys have risen as a hot topic in this field, SSH is a vital communication protocol and no way to neglect it uses from IoT devices. The research problem focusing on Raspberry Pi devices which affected by a security issue arising from the Raspbian operating system which makes generating of SSH keys weak and predictable.

The rest of this paper is organized as follows. Section 2 presents a related work for our study. The Problem and motivation is discussed in Sect. 3. In Sect. 4, the security evaluation of using different versions of Raspbian operating system and SSH server

was presented and discussed thoroughly. The experimental procedures steps and actions performed are presented in Sect. 5. Next, the discussion of the results and the findings of the experimental procedures are presented in Sect. 6. Finally, we conclude the paper with future work in Sect. 7.

2 Literature Review

Previous related works in the field of IoT, man in the middle attack and SSH have been studied to gain a full understanding of what have been conducted out in this field by previous researchers.

Xiaohong et al. [17] studied how to detect and defeat the man in the middle attack to reduce the loss under the MITM attacks since avoidance of the MITM attacks shows an incredible responsibility. They started the work by offering a defense strategy against MITM attacks, and then they show the communication among the attacker and the defender under Stackelberg security game framework and use the Strong Stackelberg Equilibrium (SSE) as the strategy for the defender. After that they implement a novel way to defeat the searching scope of calculating the optimal defense strategy. On the final step, they assess their ideal defense strategy by comparing it with non-strategic defense strategies. The correlation of the simulated results concludes for them that the suggested theoretic defense strategy exceeds than nonstrategic defense strategies in terms of reducing the total losses on MITM attacks. The limitation of the paper based on the focusing only on one singular service and assuming it applies to all other services without real examination.

Mauro et al. [18] provided a huge study of the literature on the man in the middle attack to examine and characterize the range of MITM attacks and classified them based on the position of an attacker in the network, environment of a transmission channel, and impersonation ways. After that, based on their analysis, they recognize some potential directions for future researchers based on the large number of literatures studied, and they suggest a categorization of MITM prevention mechanisms as follow: (1) Use secure interactive authentication to constantly validate endpoints of any communications channel and exchange public keys using a reliable channel. (2) Use few stable channels for checking if data has not been discredited and signify public keys by a certified authority. (3) Use certificate pinning and encrypt the communication by employing cryptography. (4) Check the conventional behavior of communicating endpoints, according to the allowed communication protocol.

Shubh and Sharma [19] analyzed the ways that man in the attacks works and they indicates the way it intercept the network to collect information even without outside party knowing it. They focus on their study at attacking SSL across HTTP which identified as HTTPS. The conclusive goal of their recommended system is to build a secure channel over a vulnerable network. They applied a combination of Diffie-Hellman and blowfish algorithm, Diffie-Hellman for key generation and blowfish for encryption which is improving the data security over SSL and HTTPS. Also they explained a scheme to strengthen the SSL by utilizing a Firefox add-on which can recognize any fraudulent SSL certificates. They conducted a real examination of a man in the middle attack against bank website by applying their proposed model and they

were successful to detect of MITM attack and their suggested recommendation shows an efficient method to avoid the MITM attack.

Yaoqi et al. [20] performed a methodical interpretation of browser cache poisoning attacks on HTTPS connections that expose the victim web sessions with the target site by poisoning the victim's browser cache wherein a web attacker completes a man in the middle attack on a user's HTTPS session and exchanges cached sources with malicious entireties. Their experimental study of such attack was held on 5 desktop browsers and 16 common mobile browsers and they noticed that the experimented browsers are extremely variable in their caching strategies for storing resources over SSL connections with fallacious certificates and they achieved that 99% of the tested browsers induced by BCP attacks to an immense amount. They presented guidelines for users and browser vendors to overcome BCP attacks. They have recorded their conclusions to browser vendors and approved the vulnerabilities of them and some of them have settled the problem based on their suggestion.

Esmaeil et al. [21] studied Brute-force attacks for SSH carried out on six separated universities campus networks by applying Honeypot techniques in attempts to obtain remote access to a system using information obtained from their SSH honeypots using the help of the tools and techniques employed. On the first phase, they used open VZ software with Kojoney honeypot and P0f fingerprinting tool to log innumerable details about the attacks. Secondly, they configured the firewall to allow the SSH service to be available on the web using its public static IP address. A Brute-force attack used to list the usernames and passwords that are publicly assigned and they successfully obtain remote access to SSH Honeypots and collect numerous data. They assessed the effectiveness of a variety of techniques intended to defend the systems against these attacks. They conclude the paper with an amazing table that recommending 17 lists for the protection of SSH servers.

Alsaadi and AlKubaisi [22] performed penetration testing on remote secure OpenSSH running on version 7.1p2 On Raspberry Pi 2 running Kali Linux version 3.2–4.4. Their study focused on the vulnerabilities discovered in exchange keys in SSH protocol which creates multiple CRLF injections and admits to conduct man in the middle attack to allow remote users bypass shell-command constraints via crafted X11 forwarding data. They approached an efficient security model based on an experimentation attack formed on various scenarios that can resolve the problems of enabling remote authentication access using SSH protocol exchange keys without hitting the encrypted protocols communications. They injected the secure shell session on port 22 of Kali Linux OS with the produced SSH keys which enabled them to achieve full entrance to device information. The limitations can be shown in missing of real testing attack on their suggest model structure and what they proposed maybe will suffer difficulties with newer versions of Raspberry Pi and SSH.

3 Problem and Motivation

Internet of Things (IOT) operates as one of the biggest potentials for human life transformation. It strongly becomes the quickest growing market but the more notable concern is that this growth is moving fast while there are major problems occurring

within it causing these devices to offer well-known vulnerabilities for hackers to exploit them. IOT devices configured through the network and SSH is essential for that but the dilemma resides on SSH problematic because the likelihood of generating weak keys. IOT devices and especially Raspberry PI devices were discovered to be generating predictable and weak keys which make these devices easy to be utilized by MiTM attack when owning the weak keys and by performing weak algorithm implementations these devices could be exploited. The motivation remains on the continuous use of SSH as communication protocol but we want to maintain its security and make sure it's secure and perfect.

4 Security Evaluation of Raspbian and SSH Various Versions Implementation

In this section, the evaluation of using various versions of Raspbian and SSH server among the different versions of the Raspberry Pi developments boards is presented. Keeping in mind that the same Raspbian image that is being used for the Raspberry Pi3 could be used directly with the Raspberry Pi 1 as most of the advancements that are presented on the boards are focused on providing higher processing speeds and extra functionalities [25].

As SSH is considered a necessary service to be used by developers to configure their Raspberry Pi developments boards, it was provided as a built-in service on the operating system image without the need to install it after starting the operating system. Raspberry Pi Debian through all its versions was implementing the OpenSSH as SSH server software [8]. The most interesting thing about the SD card that holds the operating system image is that it can be used directly with any version of the Raspberry Pi boards starting from the version 1 reaching the latest Raspberry Pi 3 Model B [9]. However, as Raspbian was being released since 2013-09-10 and at the time the first release was announced, the Raspberry Pi 2 & 3 were not yet released and thus the versions earlier than 2015-01-31 doesn't support the Raspberry Pi 2. In the same manner, the versions of Raspbian that are earlier than 2016-02-26 doesn't support Raspberry Pi 3 [23]. Through the different versions, it was noticed that the newer the version, the more packages, and services are being installed and kept with it. For example, starting from 2014-09-09 Raspbian release, multiple extra packages were built in the default image including Minecraft that is a game that most developers don't use for their IoT implementations. Another important notice for the same release that a downgrading was performed for Java from version 8 to version 7 exposing the system to an outdated packages security risk [24]. The 2015-09-25 release faced a huge transfer towards including a big number of preinstalled packages including games and utilities making the operating system more like regular Linux desktop distributions. Having various tools preinstalled on the development board may present a handy operating system for users that are using the board as their desktop and not an operating system for IoT device and again we are facing the same issue of having many preinstalled packages that could present a huge security breach [25]. The same approach was followed by the vendors by Raspberry to include more and more packages in the fresh operating system. An important notice about the 2016-09-23 release that it included a

preinstalled RealVNC that allows remote connection to the Raspberry Pi device and that could be utilized by the hackers to attack the system and obtain remote access [26]. So as a conclusion the Raspbian operating system is being developed to create a more user-friendly environment for developers that wants everything to be working properly without the need for much configurations.

As there are no specific operating systems for the specific hardware of each Raspberry Pi version since the same Raspbian SD card could be inserted into both Raspberry Pi 2 or Raspberry Pi 3 and both of them will work properly without any issues. The SSH implementation through different versions could present unique possibilities of attacks that could be adopted by the hackers. According to the CVE Details database, OpenSSH was found to be vulnerable to almost 100 published vulnerabilities that are distributed among different categories including DoS, Code Execution Overflow, Memory Corruption, Directory Traversal, Http Response Splitting, Bypass something, Gain Information, Gain Privileges [27]. Among the various versions, it was noticed that Dos to be one of the main categories that most OpenSSH related vulnerabilities are published. Having that said could present the IoT device to shut down or stopping the service causing the device to stop its intended action. This issue could create a big threat especially if the device was used for a critical function such as medical applications [28]. The second most common attack through all the versions was Bypassing one of the mechanisms that are used to start and initiate the SSH connection which could result in reducing the security as each one of the mechanisms and steps that are adopted by OpenSSH are meant to maintain the security as this is the main purpose of using SSH over the clear text telnet services [29]. Gaining the privileges was the third most common vulnerability on the OpenSSH through its all versions. Gaining privileges is an extremely dangerous issue as having a non-root user gaining higher privileges will allow him to experiment a huge amount of open possibilities to manipulate the system and obtain full access and control over the system [30]. As it is always recommended to perform the updates for all the packages and tools including OpenSSH it is worth mentioning that the released versions of the OpenSSH in 2017 were having in total around six various vulnerabilities that are distributed over various categories [29]. At the same time, the 2015 OpenSSH releases recorded no published vulnerabilities at all. This issue could be explained by the fact that the higher the version, the more options, and services are available in the implementation exposing the system to further exploitation vulnerabilities [27]. Another explanation could be that the latest versions of the OpenSSH are designed to provide backward compatibility including, for example, the support for the SSH V1 that is considered to be weak and crackable communication by the widely available tools [30].

5 Implementation and Analysis

The main focus of this research was about performing Man in The Middle (MiTM) attack to examine the security of the different variations provided by the SSH service. In Raspberry Pi, most of the time the SSH 22/tcp port is found to be open by the developers as it allows them to configure the development board through the network without the need of connecting it to a monitor. This section is divided into two main

subsections where the first one shows the examination process for the different variations of SSH and the second one shows the proper approaches for making the usage of SSH more secure.

5.1 Security Examination of Various SSH Implementations

The examination of the various implementations of SSH was performed on this research work by implementing each variation separately while examining the traffic through the Man in The Middle (MiTM) machine. Figure 1 Shows the detailed process followed in achieving this task and the rest of this subsection explains each part of the descried steps.

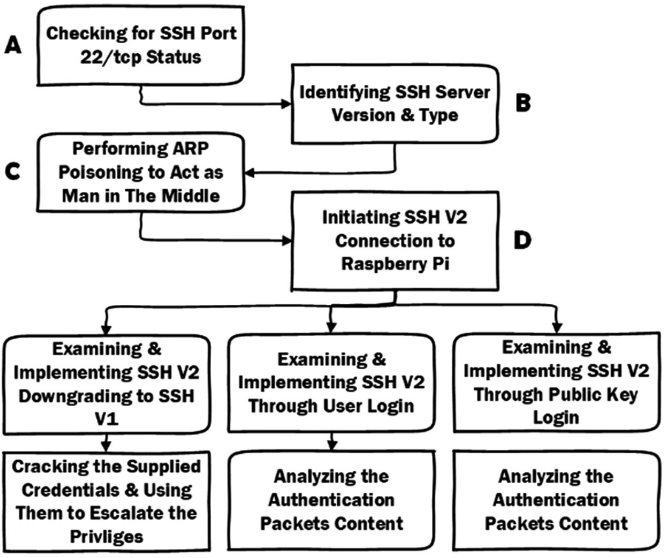

Fig. 1. Framework for performing the security examination on various SSH implementations.

(A) Checking for SSH Port 22/tcp Status
Having a port open in any target reveals a significant amount of information and possibilities for attacks [13]. As a start for our examination process the open ports in the newly installed fresh RASPBERRY PI DEBIAN JESSIE (2017-06-22) operating system was performed through n map on the Raspberry PI IP address that showed the port 22/tcp as the only open port in the device.

(B) Identifying SSH Server Version & Type
The version and the type of SSH server specifies the algorithms and procedures followed by the server to implement the service. It could also identify if it is susceptible to published vulnerabilities. Metasploit framework was used to lunch the "*auxiliary/scanner/ssh/ssh_version"* against the IP address of the Raspberry Pi the version was identified to be OpenSSH 6.7 p1 which supports SSH V1 and SSH V2.

(C) Performing ARP Poisoning to Act as Man in The Middle
Attackers tend to perform ARP poisoning to stand in between the two victims. Figure [2] shows the actual scenario that was followed in performing the MiTM attack where the Kali Linux attacker machine spoofed the IP address of both Raspberry Pi3 and the Windows 7 machines and it worked as an interceptor that sniff the traffic then pass it to its actual direction.

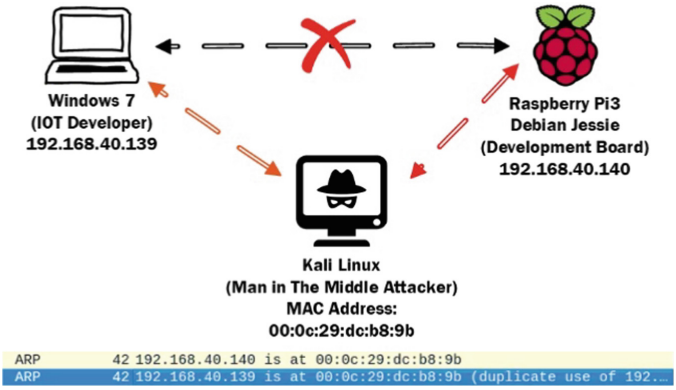

Fig. 2. Kali Linux machine acted as a MiTM to sniff the packets in both directions.

The SSH session was started in the Windows 7 machine through Putty Software to communicate with the OpenSSH server that is running in the Raspberry Pi3. At the same time the Kali Linux machine was poisoning the ARP table through the Ettercap by specifying the target 1 to be the Windows 7 IP address while the target 2 is the Raspberry Pi3 IP address. Having that set makes all the traffic between the two victims pass through the Kali Linux machine and the traffic were being analyzed through Wireshark.

(D) Initiating SSH V2 Connection to Raspberry Pi
In this part three variations of the SSH sessions were examined and analyzed. In the first scenario, the configurations for the SSH server on the Raspberry Pi were allowing the SSH V1 and SSH V2, knowing that SSH V1 is implementing weak algorithms that could be easily attacked [20]. The Kali Linux attacker machine was running Ettercap to perform ARP poisoning and implementing the ether filter that forces the SSH V2 requests to be downgraded to SSH V1 and attack the communication and reveal the user and password.

In the same way when SSH V2 communication was initiated while having the Raspberry Pi forcing only the SSH V2 and not allowing the SSH V1, Ettercap failed to downgrade or crack the credentials in both scenarios when the login is performed through user and password or through the authorized public key. The developer can select between two approaches to authenticate himself when using SSH V2. The first approach is to provide the name of one of the local users and his password and the authentication will be completed when correct credentials are provided [18]. The other

approach, is to use to have the public key of the Windows 7 machine stored in the authorized public keys list in the Raspberry Pi " ~/.ssh/authorized_keys" file. Then whenever the Windows 7 tries to authenticate itself for SSH V2 there will be no need to supply the password. The packets sequence retrieved by Ettercap MiTM scheme through Wireshark. This sequence of packets includes the negotiation about the algorithms and standards for the encryption.

As the generation process of the key pairs is based on pseudo random functions, a big risk appears for having weak predictable keys for the Raspberry Pi being generated. The issue raise there is a huge possibility of having predictable keys as the same process is followed in making the first boot process where the operating system image is copied to an SD card that is used to make the first boot with the same hardware making the pseudorandom functions behavior predictable and thus exposing the SSH V2 to the MiTM attacker that could launch similar attack to the one performed on SSH V1.

5.2 Hardening SSH Implementations

The communication through SSH could be hardened to ensure that it is not easily breakable or hackable. Figure [3] shows the considered three main sources of threat against SSH communication and the proposed solution.

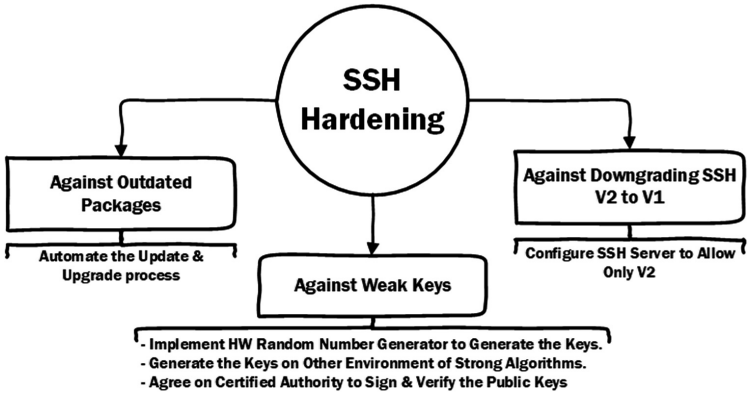

Fig. 3. Main threat sources to SSH communication and their proposed solutions.

Keeping in mind that each outdated package that is installed in the Raspberry Pi machine could present a big threat due to possible vulnerabilities. As an example, to that, an outdated version of OpenSSH could support only SSH V1 which is adopting weak algorithms that could be attacked easily making the communication not secure. Automating the update and upgrade process in the machine and using patch management tools could limit such type of threats. To prevent the downgrading of SSH V2 to SSH V1, the configuration file on the Raspberry Pi in "/etc./ssh/sshd_config" could be check to make sure only Protocol 2 is allowed. As the implementation of the Raspberry Pi OS makes it generating weak predictable keys, it is recommended to use

well tested tools on other machines such as Putty in Windows environment to generate the key pair then add them to the Raspberry Pi SSH keys. Another solution to the weak keys issue, could be implementing an indirect way to use hardware random numbers generators through the following steps:

- Obtain the Raspberry Pi chipset number.
- Load the chipset related random number generator module.
- Install random number generator package and activate it.
- Now the entropy will be having a larger size of pool making the generation pseudo random numbers process more efficient.
- Remove old keys that were generated on the first boot and regenerate a new strong pair of keys and use them.

Even when the presence of the weak key, a good solution could be implemented to have a certified authority that signs the public keys of both parties to ensure that they are not the public key of the MiTM attacker machine. In that way, both parties don't start the SSH communication unless they verify that the public keys are genuine and not belonging to unknown party.

6 Results and Discussion

Upon the examination of the obtained results from the implementation and experimental procedures, a set of outcomes were drawn that could be valid on most IoT based development boards which share the same nature of the Raspberry Pi. The discussion of the findings is presented in the rest of this section.

When an attacker is performing network scanning to identify the live host, the Raspberry Pi devices could be identified from their OUI/MA-L part of the networking interfaces MAC addresses. The OUI/MA-L for the Raspberry Pi Foundation is B8-27-EB (hex) based on experimentation and the OUI standards that are maintained by IEEE. Once a Raspberry Pi device is discovered and an intensive port scanning is performed to it, it is expected to find the SSH 22/tcp port to be the only port that is open. Accordingly, attackers will be directing most of the time their attacks on this port and service raising the need to secure it as much as possible. In some other cases in addition to the SSH server being running on the Raspberry PI, a WEB server is the second most probably service that could be running as well to allow the communication and control for the users through WEB portals that are based on the Raspberry PI.

As operating system installation was the main issue that caused the Raspberry Pi to generate weak keys that are predictable, the same threat could be valid to other IoT development boards. Most development boards, uses an SD card that is having an image of the operating system to run its hardware leading to similar environments that could result in predictable keys. In other words, if two development boards with the same hardware and were booted from exactly similar SD card images while not having a big entropy pool for the random numbers generation process as the first boot then the big chance of generating similar predictable keys could be happening. Accordingly, the generated keys from the first boot must be regenerated with strong keys before stating to use the SSH service.

As it was proved earlier that Ettercap was successful in performing MiTM attack and crack the SSH V1 communication revealing the user and password in plain text, it is extremely important to prevent SSH server from allowing this version of SSH as it will be exposing the communication and credentials to the attackers.

Having an attacker granted an SSH session to the device even if was with a user that is having limited privileges could expose the system to an endless exposure. Many exploits are available that allow privilege escalation and many Linux versions are vulnerable to them. An example of privilege escalation was performed on the Raspberry Pi 3 device through the DirtyCow exploit that changes the root password from when running the DirtyCow script file from any non-root user.

When an attacker is performing MiTM attack on a SSH session, he will be providing his own public key so that the other side used it to encrypt the data instead of the destination public key. It is recommended that the public key of the administrator machine being hardcoded or another trusted certification authority is involved that signs the public keys to verify their authenticity.

7 Conclusions and Future Work

IoT devices draw a clear plan towards the future, which requires us to measure all the risks that can be exposed to these devices and propose solutions that will improve the operation of this process toward the best use without compromising privacy and security of people. In this paper, we have examined the security of SSH on Raspberry Pi Debian Jessie device and we discovered to be generating weak key pair in the first boot due to not implementing hardware number generators. It's recommended to change the default first boot key pair on any device since the randomization process and pool were not yet efficient leading to weak and predictable keys. Based on that we proposed various solutions to harden SSH on its various communication scenarios since we cannot abandon it due to its importance for communicating with the device. Thus our proposed solutions could be summarized in four points through (1) checking the configuration file on the Raspberry Pi in "/etc./ssh/sshd_config" and guarantee only Protocol 2 is allowed. (2) Using well-tested tools on other machines such as Putty in Windows environment to generate the key pair then add them to the Raspberry Pi SSH keys. (3) Implementing an indirect way to use hardware random numbers generators through suitable suggested steps. (4) Having a certified authority that signs the public keys of both parties to ensure that they are not the public key of the MiTM attacker machine.

As a future work, a controlled experiment could be conducted on multiple Raspberry Pi development boards to examine and predict the behavior of the generated first boot key pairs. After obtaining the behavior that predicts the possible key pairs, this behavior can be used to test other multiple Raspberry Pi development boards but after implementing our proposed approach in adopting hardware random number generator. As an extension to that, a script could be written to perform MiTM attack between two targets that are communicating through SSH and use the predicted key pairs to crack the communication.

References

1. Ramirez, J., Pedraza, C.: Performance analysis of communication protocols for internet of things platforms. In: 2017 IEEE Colombian Conference on Communications and Computing (COLCOM), pp. 1–7 (2017)
2. Junaid, M., Shah, M.A., Satti, I.A.: A survey of internet of things, enabling technologies and protocols. In: 2017 23rd International Conference on Automation and Computing (ICAC), pp. 1–5 (2017)
3. Pan, J., McElhannon, J.: Future edge cloud and edge computing for internet of things applications (2017)
4. Hassan, R., Jubair, A.M., Azmi, K., Bakar, A.: Adaptive congestion control mechanism in CoAP application protocol for internet of things (IoT). In: 2016 International Conference on Signal Processing and Communication (ICSC), pp. 121–125 (2016)
5. Lei, W., Xu, L.: Research and implementation of access control model of internet of things. In: 2016 5th International Conference on Computer Science and Network Technology (ICCSNT), pp. 102–106 (2016)
6. Ren, Z., Liu, X., Ye, R., Zhang, T.: Security and privacy on internet of things. In: 2017 7th IEEE International Conference on Electronics Information and Emergency Communication (ICEIEC), pp. 140–144 (2017)
7. Prabavathy, S., Sundarakantham, K., Shalinie, S.M.: Decentralized secure framework for social collaborative internet of things. In: 2017 Fourth International Conference on Signal Processing, Communication and Networking (ICSCN), pp. 1–6 (2017)
8. Marot, J., Bourennane, S.: Raspberry Pi for image processing education. 2017 25th European Signal Processing Conference (EUSIPCO), pp. 2364–2366 (2017)
9. Bhave, S., Tolentino, M., Zhu, H., Sheng, J.: Embedded middleware for distributed raspberry Pi device to enable big data applications. In: 2017 IEEE International Conference on Computational Science and Engineering (CSE) and IEEE International Conference on Embedded and Ubiquitous Computing (EUC), vol. 2. pp. 103–108 (2017)
10. Sanada, A., Nogami, Y., Iokibe, K., Khandaker, M.A.A.: Security analysis of raspberry Pi against side-channel attack with RSA cryptography. In: 2017 IEEE International Conference on Consumer Electronics - Taiwan (ICCE-TW), pp. 287–288 (2017)
11. Tavade, T., Nasikkar, P.: Raspberry Pi: data logging IOT device. In: 2017 International Conference on Power and Embedded Drive Control (ICPEDC), pp. 275–279 (2017)
12. Dowling, S., Schukat, M., Melvin, H.: A ZigBee honeypot to assess IoT cyberattack behaviour. In: 2017 28th Irish Signals and Systems Conference (ISSC), pp. 1–6 (2017)
13. Eigner, O., Kreimel, P., Tavolato, P.: Detection of man-in-the-middle attacks on industrial control networks. In: 2016 International Conference on Software Security and Assurance (ICSSA), pp. 64–69 (2016)
14. Dowling, S., Schukat, M., Melvin, H.: Using analysis of temporal variances within a honeypot dataset to better predict attack type probability. In: Proceedings of the IEEE World Congress on Internet Security, (WorldCIS 2016) (2017)
15. Song, I.-A., Lee, Y.-S.: Improvement of key exchange protocol to prevent man-in-the-middle attack in the satellite environment. In: 2016 Eighth International Conference on Ubiquitous and Future Networks (ICUFN), pp. 408–413 (2016)
16. Saqib, N.: Key exchange protocol for WSN resilient against man in the middle attack. In: 2016 IEEE International Conference on Advances in Computer Applications (ICACA), pp. 265–269 (2016)
17. Li, X., Hao, J., Feng, Z., An, B.: Optimal personalized defense strategy against Man-In-The-Middle attack. vol. 2 (2017)

18. Conti, M., Dragoni, N., Lesyk, V.: A survey of man in the middle attacks. IEEE Commun. Surv. Tutorials **18**(3), 2027–2051 (2016)
19. Shubh, T., Sharma, S.: Man-In-The-Middle-Attack prevention using HTTPS and SSL, vol. 5, no. 6, pp. 569–579 (2015)
20. Chen, Y., Dong, X., Saxena, P., Mao, J., Liang, Z.: Man-in-the-browser-cache: persisting HTTPS attacks via browser cache poisoning. Comput. Secur. **55**, 62–80 (2015)
21. Kheirkhah, E., Amin, S., Sistani, H., Acharya, H.: An experimental study of SSH attacks by using Honeypot Decoys, vol. 612, pp. 5567–5578 (2013)
22. Alsaadi, H., AlKubaisi, M.: Penetration Testing of Remote Secure OpenSSH On Raspberry Pi 2. Unpublished manuscript (2016)
23. De Luca, G.E., Carnuccio, E.A., Garcia, G.G., Barillaro, S.: IoT fall detection system for the elderly using intel Galileo development boards generation I. In: 2016 IEEE Congreso Argentino De Ciencias De La Informática y Desarrollos De Investigación (CACIDI), pp. 1–6 (2016)
24. Valverde, M.P., González, J.: A software controlled hardware acceleration architecture for image processing using an embedded development board. In: 2016 IEEE 36th Central American and Panama Convention (CONCAPAN XXXVI), pp. 1–5 (2016)
25. Mischie, S., Muntean, A.: Distance estimation through stereoscopy using BeagleBoneBlack and RaspberryPi. In: 2017 International Symposium on Signals, Circuits and Systems (ISSCS), pp. 1–4 (2017)
26. Sukvichai, K., Wongsuwan, K., Kaewnark, N., Wisanuvej, P.: Implementation of visual odometry estimation for underwater robot on ROS by using RaspberryPi 2. In: 2016 International Conference on Electronics, Information, and Communications (ICEIC), pp. 1–4 (2016)
27. Coonjah, I., Catherine, P.C., Soyjaudah, K.M.S.: Performance evaluation and analysis of layer 3 tunneling between OpenSSH and OpenVPN in a wide area network environment. In: 2015 International Conference on Computing, Communication and Security (ICCCS), pp. 1–4 (2015)
28. Coonjah, I., Catherine, P.C., Soyjaudah, K.M.S.: A VPN framework through multi-layer tunnels based on OpenSSH. In: International Conference on Computing, Communication & Automation, pp. 1395–1401 (2015)
29. Studiawan, H., Pratomo, B.A., Anggoro, R.: Clustering of SSH brute-force attack logs using k-clique percolation. In: 2016 International Conference on Information & Communication Technology and Systems (ICTS), pp. 39–42 (2016)
30. Sadasivam, G.K., Hota, C., Anand, B.: Classification of SSH attacks using machine learning algorithms. In: 2016 6th International Conference on IT Convergence and Security (2016)