Vulnerabilities in IoT Devices with Software-Defined Radio

Phan Duy Hung
FPT University
Hanoi, Vietnam
e-mail: hungpd2@fe.edu.vn

Bui Trong Vinh
FPT University
Hanoi, Vietnam
e-mail: vinhbtmse0094@fpt.edu.vn

***Abstract*—Advances in networking and semiconductor technologies, along with the ever-widening grid of inter-connected and computationally capable products have promoted the development of the Internet of Things (IoT). This development naturally poses more and more complex security challenges. One of the key attributes of IoT is that it makes heavy use of wireless communications to allow for mobility and ease of installation. It is important to note this is not just Wi-Fi, but all manner of other Radio Frequency (RF) protocols: Bluetooth, BTLE, ZigBee, Z-Wave, etc. The increasing ubiquity of such devices and networks promises to make life easier (smart locks, smart bulb, smart home appliances…), however manufacturers often overlook the security in the implementation of these RF communication systems. This paper uses the Software Defined Radio (SDR) to study vulnerabilities in IoT devices using an unknown RF protocol as the analyzing frequency, demodulation and decoding RF signals used in the wireless IoT devices, jamming the target and replaying radio packets. A number of schemes have been proposed to mitigate such attacks. This study also provides an empirical environment sufficient to prove the exploit of that vulnerabilities.**

Keywords-IoT; vulnerability; software-defined radio; unknown RF protocol

I. INTRODUCTION

A. *Problem and Motivation*

The Internet of Things (IoT) includes a wide variety of devices and diverse applications, which call for different deployment scenarios. There are many market segments and verticals poised to drive IoT growth, some of them are: consumer goods, smart transportation, energy distribution, smart city, distribution and logistics, industrial and manufacturing, smart home, eHealth. A general architecture of the IoT can be described by defining three generic layers: Physical or Perception layer, Network layer and Application layer. Within any IoT architecture, threats are everywhere and vulnerabilities can be exploited for a number of malicious purposes (Fig. 1).

To allow all the expected billion of devices to communicate with each other, a communication technology is needed. Currently, there are plenty of wireless communication standards and specifications and each of them have strengths for certain applications. State of the art classic communication networks are being adapted to support IoT characteristics. In order to group the surveyed technologies, grouping criteria must be defined. Most of the technologies are based on wireless communications. So, a first group can be defined for the wired networking solutions in order to differentiate them. For the wireless solutions, the following figure show their range versus data rate (Fig. 2).

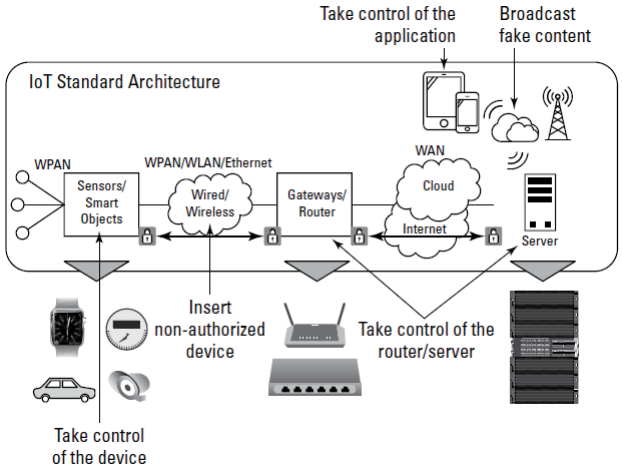

Figure 1. Threats in IoT.

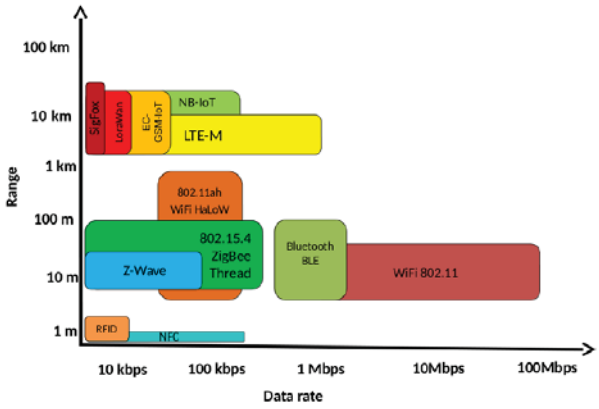

Figure 2. Threats in IoT.

It was seen during the project that grouping them by its range is an adequate criteria. It was found that all the technologies in a certain range tend to implement similar solutions. And that these solutions are quite different from those implemented in other ranges. For instance, ZigBee (short-range) is quite different than technologies such as RFID (very short-range) and NB-IoT (long-range) but have lots of similarities with technologies such as Z-Wave and Thread.

978-1-7281-1322-7/19/$31.00 ©2019 IEEE

Although wireless sensor connections offer several ways to increase our productivity in many fields such as smart home, smart production or smart transportation, it also introduces some risks. The usage of a wireless physical communication, which allows attackers easier interception of communications, together with the Internet of Things (IoT) or Web of Things (WoT) also leads to unprecedented opportunities for attackers to reveal confidential information and to manipulate data. It is crucial to find efficient and effective methods to counteract such attacks. Otherwise, all the benefits of the IoT will be forfeited. In [1] Daniel, S. et al present the Intercepting the Sector Sweep to Launch Man-in-the-Middle Attacks on Wireless IEEE 802.11ad Networks. Lucia, G. et al evaluate the resistance to jamming and casual interception in mobile wireless networks [2]. Pieter, R. give the injection attacks on 802.11n MAC frame aggregation [3]. The authors in [4,5] exploit security issues in the WSN. Boris, D. et al. attack on physical-layer identification [6]. Some studies focus on special protocols such as: How privacy leaks from Bluetooth mouse in [7], man-in-the-middle attacks on Secure Simple Pairing in Bluetooth 5.0 and its countermeasure in [8]. Finally, in [9], the authors explore the security issues in ZigBee networks.

Along with these related work, this paper focuses on exploiting all the most common attacks on IoT devices using an unknow RF protocol and proposes some schemes for mitigating those attacks. Additionally, this paper includes an empirical environment sufficient to prove the exploit of that vulnerabilities, several tools for security researchers and professionals to consider to help them be most effective when working with wireless protocols.

The remainder of the paper is organized as follows: Section 2 describes the research environment. Vulnerability analyzing will be provided in Section 3. Furthermore, Section 3 also gives a number of schemes for mitigating those attacks. Section 4 concludes the paper and provide some of the author's perspectives.

II. Research Environment

To set up research environment, some Software Defined Radio (SDR) tools are used. An SDR is a versatile piece of hardware that can change reception and transmission profiles based on software configuration. An SDR are available in half-duplex (only reception or transmission at a time) or full-duplex mode (reception and transmission simultaneously). As the name suggests, the radio in this case is software defined, which means that the functionality, or the action that the radio performs, can be changed and modified based on our requirements. This is unlike traditional radios, where a particular device served a single purpose based on the hardware design present in it.

A. Checklist

Here is a list of tools to use for research in this paper:

1. GNURadio
2. GQRX
3. HackRF One (1 device)
4. Arduino Uno R3 (2 boards)
5. RF 433MHz Transmitter (1 module)
6. RF 433MHz Receiver (1 module)
7. Test board (2 board), LED (1), hook-up wires.

The step-by-step guides to install the tools 1 and 2 from the source can be found on the following links:

- GQRX [10]: https://github.com/csete/gqrx
- GNURadio [11]: https://wiki.gnuradio.org/index.php/InstallingGRFromSource

The HackRF One device was purchased from Adafruit.com [12]. The remaining components are purchased from banlinhkien.com [13].

B. GNURadio for Radio Signal Processing

GNURadio is an open source software development kit to handle digital and analog signal processing. It also supports a wide range of SDR hardware tools such as RTL-SDR, HackRF, USRP, and more, and includes a huge variety of radio processing blocks and applications which can be used to process the data.

GNURadio, simply put, is an open source tool which allows us to work with various radio components. Various input sources, processing blocks, and output forms can be utilized. The GNURadio applications can also be built using Python scripting, which internally call the C++ signal processing code of GNURadio and give the desired output. GNURadio Companion is a graphical utility that comes along with the GNURadio toolkit which allows building flow graphs using the underlying GNURadio components.

C. Gqrx

Gqrx is an open source software defined radio receiver (SDR) powered by the GNU Radio and the Qt graphical toolkit. Gqrx supports many of the SDR hardware available, including Airspy, Funcube Dongles, rtl-sdr, HackRF and USRP devices. Gqrx offers main features like:

- Discover devices attached to the computer.
- Change frequency, gain and apply various corrections (frequency, I/Q balance).
- FFT plot and waterfall.
- Record and playback audio to / from WAV file.
- Spectrum analyzer mode where all signal processing is disabled.

D. HackRF One

The HackRF One (Figure 3) is a Software Defined Radio peripheral capable of transmission or reception of radio signals from 1 MHz to 6 GHz. Designed to enable testing and development of modern and next generation radio technologies, the HackRF One is an open source hardware platform that can be used as a USB peripheral or programmed for stand-alone operation. These SDRs can be used to analyze the signal transmitted between IoT devices and transmit rogue messages.

Figure 3. A HackRF one device.

E. RF Transmitter/Receiver Circuits

In this system, the transmitter 2 seconds will send an "OK!" String. The receiver receiving the message will light the LED for 1 second and then turn it off. A very simple wiring diagram and source code is included below. Fig. 4 is the transmitter circuit diagram and Fig. 5 is the receiver circuit diagram.

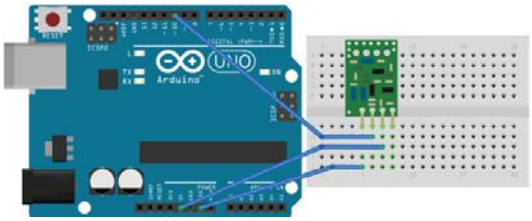

Figure 4. Transmitter circuit.

Source code of the transmitter:

```
#include <RH_ASK.h>
#include <SPI.h>
RH_ASK driver;
void setup()
{
    Serial.begin(9600);   // Debugging only
    if (!driver.init())
         Serial.println("init failed");
}
void loop()
{
    const char *msg = "OK!";
    driver.send((uint8_t *)msg, strlen(msg));
    driver.waitPacketSent();
    delay(2000);
}
```

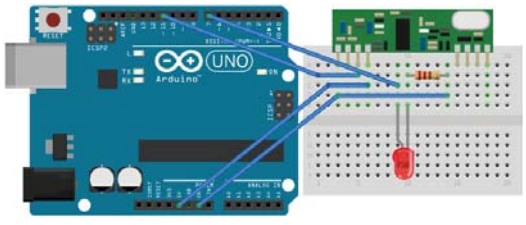

Figure 5. Receiver circuit.

Source code of the receiver:

```
#include <RH_ASK.h>
#include <SPI.h>
RH_ASK driver;
const int ledPin = 7;
void setup()
{
    pinMode(ledPin, OUTPUT);
    Serial.begin(9600); // Debugging only
    if (!driver.init())
         Serial.println("init failed");
}
void loop()
{
    uint8_t buf[3];
    uint8_t buflen = sizeof(buf);
    if (driver.recv(buf, &buflen)) // Non-blocking
    {
     if((buf[0]=='O')&&(buf[1]=='K')&&(buf[2]=='!'
)) {
        digitalWrite(ledPin, HIGH);
        delay(1000);
        digitalWrite(ledPin, LOW);
      }
    }
}
```

III. Vulnerability Analyzing Using SDR

In this section, vulnerabilities of IoT Devices using an unknown RF protocol are described and a number of schemes have been proposed to mitigate such attacks.

A. Identifying the Frequency of a Target

One of the most important pieces of analysis when starting any IoT device radio analysis is to identify its operating frequency.

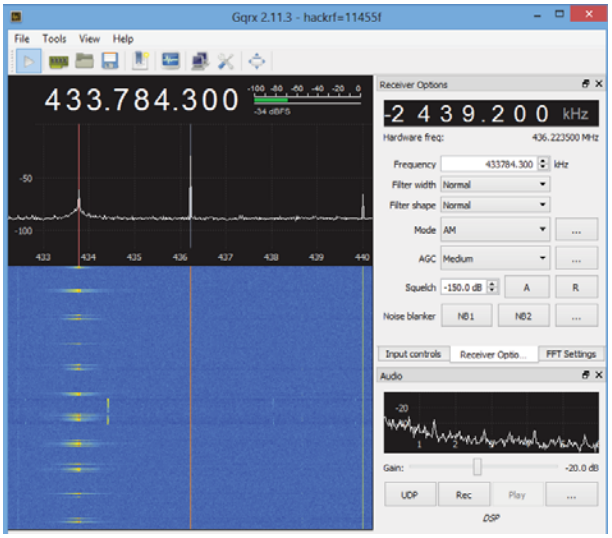

Figure 6. Frequency spectrum.

This information can sometimes be publicly available such as with the FCC ID information or on the device's website or on community forums. In case it is not present or not easily obtainable, an SDR tool such as HackRF One is used, which will enable monitoring a wide range of frequency spectrums covering the frequency on which the

device would most often be operating on. The software utility used to look at the frequency spectrum is GQRX. In this research, it is known that the device operates on a 433 MHz frequency, hence around the 433 MHz range in GQRX is where the spikes can be identified.

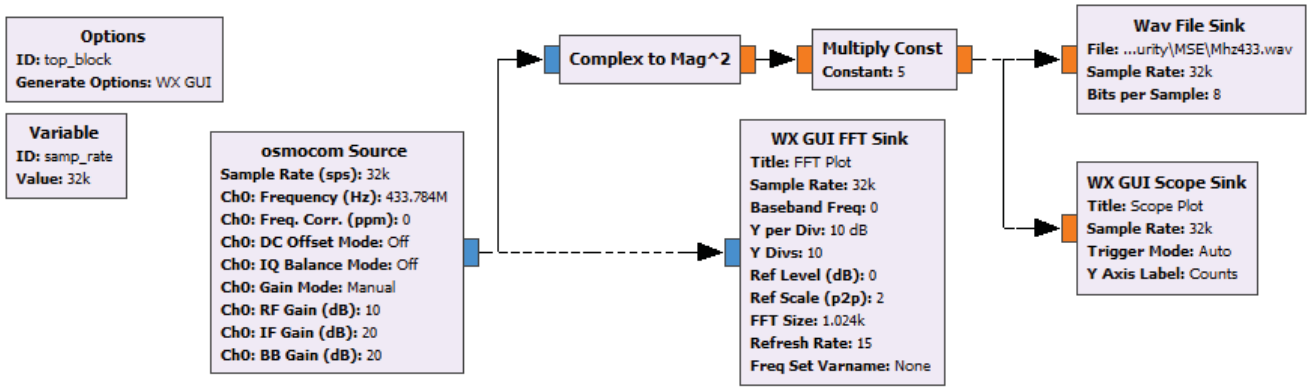

Figure 7. Visualization of dataset after being transposed.

As shown below in Fig. 6, the highest spikes occur at 433.784.300 Hz, which is the frequency for RF module.

B. *Decode Data Unknown RF Protocol*

To decode the signal data, the GNURadio companion tool, in which various radio blocks can be implemented, is used to process the data.

Fig. 7 is the block diagram for decoding the signal data. The signal emitted by the transmitter will be captured by the HackRF One controlled by the osmocom Source block. Received signals are displayed as spectra to be observed and recorded in a wav format file for several cycles, e.g. 10 seconds.

There are a few things to note in this flowgraph:

- Frequency is 433.784.300 Hz
- Sampling rate is 32k Hz
- The file name is 'Mhz433.wav'
- A "waterfall graph" of what it is recording is shown.

To decode the signal stored in the wav file, it is open in Audacity, which is a utility used for modifying and analyzing audio files and raw captures. As shown in the figure 8, it looks like an On-Off-Keying (OOK). Here, the shorter pulse gap represents a “digital” one and longer one represents a “digital” zero.

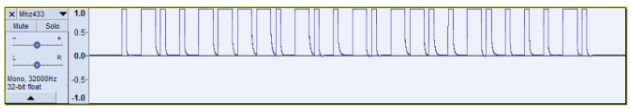

Figure 8. Decoding signal data.

Finally, the signal is segregated into 8-bit blocks and converted into text (decimal to ASCII). Evidently, the secret string has been successfully extracted, which in this case is the word “OK!”.

01001111 01001011 00100001
O K !

C. *Replaying Radio Packets*

Another form of attack using SDRs is the Replay Attack. It works by simply recording a signal, and then rebroadcasting it.

Now, simply by broadcasting the file recorded in the previous section, a replay attack is executed. A flowgraph for transmitting each of the messages is shown below (Fig. 9).

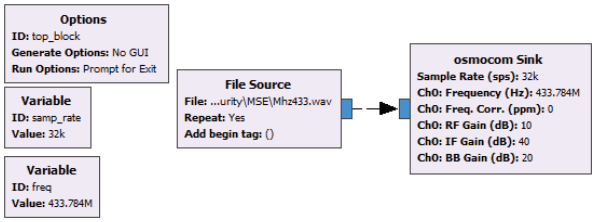

Figure 9. Flowgraph to broadcast the signal data.

As a result, after replaying, the LED in the receiver circuit changes state as usual.

D. *Jamming*

Jamming or a "denial-of-service" RF attack is easy accomplished. It will disrupt radio communication between Transmitter and Receiver.

Jammers are likely to cause problems with remotely operated aerial drones, communications systems and other radio-based functions. They also have a limited potential, since they must be set to operate within the right frequency.

The RF jammers operate by transmitting a high-power signal on the same frequency as the targeted device. The signal collides with signals sent to and from the targeted device, rendering it useless.

In the present example we have repeatedly transmitted a sinusoidal signal at the same frequency (433.784.300 Hz) to the receiver from the transmitter to the receiver. (Fig. 10)

As a result of the jamming, the lamp does not switch status.

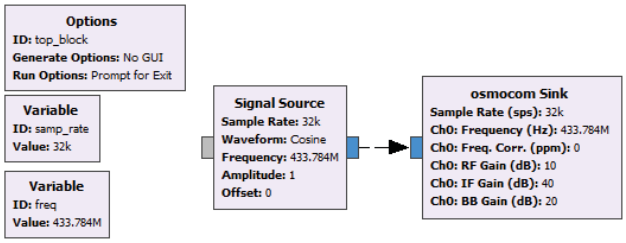

Figure 10. Flowgraph to jam the signal data.

E. Mitigation for RF Attacks

Defense against RF attacks is a combination of the following:

- Tightening: try to use directional radio beams. The transmitter will send a non-isotropic signal, much stronger in one direction than in any other; the receiver will also concentrate its reception ability on this specific direction. This method is a good prevention for sniffing and jamming.
- Frequency hopping: switch frequencies over a large range of possible frequencies. Sender and receiver agree upon the sequence of frequencies that they will use. This relies on the idea that it is much harder to jam or to sniff a large spectrum than a single well-defined frequency.
- Using session ID, One-time passwords or Timestamps combined with Advanced Encryption Standard (AES), RSA encryptions are good solutions to sniffing, replaying.
- Power: The nature of jamming is that the noise is much larger than the signal. Therefore if we increase the power of the transmitter, it will take more noise to drown it out. Of course, this increases energy consumption and heat dissipation.
- Locate the source of the noise: a jammer is an active attacker; it emits a strong signal. This allows for tracking of the jammer and providing appropriate processing.

The method chosen will depend on the required level of security, hardware capabilities (antenna, speed, memory, encryption, modulation, etc).

IV. Conclusion and Perspectives

The article fully presents all the unknown RF protocol vulnerabilities using Software Defined Radios.

Tools such as GQRX, GNURadio, and HackRF are fully utilized. These concepts, even though covered briefly in this paper, will prove useful for a lot of practical purposes.

GQRX was used for identifying the frequency of a Target. We have used GNURadio in most of our IoT penetration testing engagements before, to decode radio communication being performed, or to reverse engineer an unknown protocol, and used tools as HackRF for reply attack, jamming IoT devices.

Finally, based on the vulnerability surveyed, the paper pointed out the defensive and security solutions for unknow RF protocol as tightening, frequency hopping, using advanced encryption with a session ID and a component number, increasing transmit power, locating the source of the noise.

The construction of the offensive or defensive attack solution is always parallel and extremely complex. This paper is the initial results for us to continue to study improvements on known protocols.

References

[1] Daniel, S., Yimin, Y., Matthias, H. Beam-Stealing: Intercepting the Sector Sweep to Launch Man-in-the-Middle Attacks on Wireless IEEE 802.11ad Networks. June 2018 WiSec '18: Proceedings of the 11th ACM Conference on Security & Privacy in Wireless and Mobile Networks [doi>10.1145/3212480.3212499].

[2] Lucia, G., Gian-Luca, D. R., Andrea, M. et al. Evaluating resistance to jamming and casual interception in mobile wireless networks. October 2012 MSWiM '12: Proceedings of the 15th ACM international conference on Modeling, analysis and simulation of wireless and mobile systems [doi>10.1145/2387238.2387265].

[3] Pieter, R., Peter, Q., Wim, L. Injection attacks on 802.11n MAC frame aggregation. WiSec '15 Proceedings of the 8th ACM Conference on Security & Privacy in Wireless and Mobile Networks [doi>10.1145/2766498.2766513].

[4] Yanqiang, S., Xiaodong, W., Xingming, Z. Jamming attack in WSN: a spatial perspective. UbiComp '11 Proceedings of the 13th international conference on Ubiquitous computing [doi>10.1145/2030112.2030214].

[5] Brian, Y., Neal W. 2018. Assessing security risk for wireless sensor networks under cyberattack. Proceedings of the Annual Simulation Symposium (ANSS '18). Society for Computer Simulation International, San Diego, CA, USA, Article 1, 12 pages.

[6] Boris, D., Heinrich, L., Srdjan, C. et al. 2010. Attacks on physical-layer identification. The third ACM conference on Wireless network security (WiSec '10). ACM, New York, NY, USA, 89-98 [doi>/10.1145/1741866.1741882].

[7] Xian, P., Zhen, L., Aniket, P. et al. How privacy leaks from bluetooth mouse?. CCS '12 Proceedings of the 2012 ACM conference on Computer and communications security [doi>10.1145/2382196.2382309].

[8] Da-Zhi, S., Yi, M., Willy, S. 2018. Correction to: Man-in-the-middle attacks on Secure Simple Pairing in Bluetooth standard V5.0 and its countermeasure. Personal Ubiquitous Comput. 22, 1 (February 2018), 69-69 [doi>10.1007/s00779-017-1085-2].

[9] Lindsey, N. M., Todd, R. A., McDonald, J. T. Exploring security in ZigBee networks. CISR '14 Proceedings of the 9th Annual Cyber and Information Security Research Conference [doi>10.1145/2602087.2602090].

[10] https://github.com/csete/gqrx

[11] https://wiki.gnuradio.org/index.php/InstallingGRFromSource

[12] https://www.adafruit.com/product/3583

[13] http://banlinhkien.vn/.