An Investigation of Bluetooth Security Vulnerabilities

Peter Cope, Joseph Campbell
Computer Science Department
New York Institute of Technology
Old Westbury, NY, USA
{pcope, jcampb01}@nyit.edu

Thaier Hayajneh
Computer and Information Science Department
Fordham University
New York, NY, USA
thayajneh@fordham.edu

Abstract—**As Bluetooth technology has evolved and improved over the years, it has gained widespread acceptance and is increasingly found in many aspects of everyday life. It's convenient and easy to use, but it also has security flaws which make it vulnerable to attacks. In this paper, we discuss and demonstrate some of the tools and techniques that are currently available to attackers to exploit the vulnerabilities in Bluetooth. We also discuss some of the techniques to mitigate those risks to protect data and devices. Yet despite its world-wide acceptance and continued proliferation, as we have learned through our research and analysis, security vulnerabilities in Bluetooth still persist.**

Keywords—**Bluetooth; 802.15.1; Vulnerability analysis; Security; Ubertooth One**

I. INTRODUCTION

Bluetooth has become one of the most widely used technologies in use today, and is the standard for short-range wireless communications that allows devices to connect and exchange information. It can be found at homes in devices such as cell phones, headsets, speakers, printers, keyboards, automobiles, and medical devices. Increasingly it is being used in automating smart homes, providing monitors and controls for Internet-of-Things (IoT) devices such as lights, thermostats, door locks, appliances and even security systems.

The name Bluetooth came from Harald Blaatand "Bluetooth", who was king of Denmark in the 10th century. King Harald united the people of Scandinavia, and the name was chosen to signify the uniting of personal wireless electronic devices.

Sven Mattisson graduated as an engineer in 1979 and began working in the Faculty of Engineering at Lund University. Soon after he became an exchange student at the California Institute of Technology, where he worked on his PhD Thesis in which he developed a circuit simulator that would reproduce analog behavior in an integrated circuit. In 1995 he was hired by Ericsson Mobile Communications, where he started working on short range radio links with low output. The company was looking to develop mobile communications that would allow a device to communicate without having to be connected by cables. The technology was initially called MCL which stood for Multi-Communicator Links.
Ericsson soon realized that they needed more help developing the technology, and in 1997 Intel came on board. Jim Kardach from Intel joined Mattison, and together their vision went beyond the telephone. They wanted the technology to be available to other wireless devices as well. It was finally launched in 1998.

Bluetooth wasn't immediately ready for mass production, and it took a while for people to warm up to it. Some dismissed it due to the low data transfer rate of only 721 kbps. But gradually the technology started to make inroads, and within 10 years there were three output specifications and data transfer rates that exceeded 20 Mbps. In 2000 the first Bluetooth headset appeared on the market. In March 2002, it was ratified by IEEE for 802.15.1. Almost 3,000 companies have jumped on board and began integrating this technology into their devices. There are about ten patents that comprise the Bluetooth technology; and most of them are owned by Ericsson. There are future plans for increasing Bluetooth speeds that could attain several hundred Mbps [1].

In this paper, we discuss and demonstrate some of the tools and techniques that are available for attackers to exploit the vulnerabilities in Bluetooth. We also investigate some of the techniques to mitigate those risks to protect data and devices. The remaining sections of paper are organized as follows. Section 2 presents an overview and background information about Bluetooth. The security of Bluetooth is analyzed in Section 3 and Bluetooth vulnerabilities are discussed in Section 4. In Section 5 we show our performance analysis and results. Mitigation techniques are presented in Section 6 where open issues are discussed in Section 7. Finally, Section 8 concludes the paper.

II. OVERVIEW OF BLUETOOTH

Bluetooth operates in the unlicensed Industrial, Scientific and Medical (ISM) radio frequency (RF) 2.4 GHz spectrum. There are 3 classes of devices offering 3 connectivity ranges. Class 1 devices transmit at 100 mW and offer a range of 100 meters. Class 2 devices, the most common, transmit at 2.5 mW and

978-1-5090-4228-9/17/$31.00 ©2017 IEEE

have a range of 10 meters. Class 3 devices transmit at 1 mW and have a range of about 1 meter. A key advantage of Bluetooth is that it can transmit both voice and data simultaneously. It supports asynchronous (data) links and synchronous (audio) links, and error handling is provided for the asynchronous links through re-transmission of packets [2]. One of the disadvantages of Bluetooth is that it shares the 2.4 GHz radio frequency spectrum with many consumer appliances, i.e. microwaves, baby monitors and cordless phones. This creates the possibility of interference with those devices when they are in use [3][17].

A piconet is a group of 2 or more Bluetooth devices that communicatee with one another. The connection between a cell phone and a wireless headset is an example of a simple Bluetooth piconet. Connectivity in a piconet is spontaneous in an ad hoc manner. One device is designated as the master and all other devices are known as slaves. Most Bluetooth devices can operate as either a master or a slave. The master is the one that initiates the piconet by first searching for devices within range that are in discoverable mode, at 1.28 second intervals. When it finds a device that it wants to join the piconet, the master will send an invitation to that device. Devices can belong to more than one piconet, and the combination of 2 or more piconets forms a scatternet. While a device can only be a master in 1 piconet, it can also be a slave in another piconet. Each active slave is assigned a unique active member address, AM_ADDR, by the master. It is possible to have more than 7 slaves registered by the master, but a maximum of 7 can be active at one time. The exception is Bluetooth LE, which can have an unlimited number. The other devices will be "parked" and may be invited by the master to become active at a later time. Devices that are not part of a piconet are considered stand-by [2].

Every Bluetooth device has a unique 48 bit address that is assigned by the manufacturer. This Bluetooth device address is referred to as the BD_ADDR. In addition to the BD_ADDR, Bluetooth devices have 3 entities that are used for security purposes. A 128-bit private authentication key is used for authentication. A private encryption key is used for encryption, and it varies in length from 8 to 128 bits. A random number (RAND) is a 128 bit pseudo-random number that the Bluetooth device generates. The device has a service database and a device database for storing security information. The security manager uses those databases to control authentication, authorization and encryption. When a device receives a request from another device, it uses the RFCOMM protocol to communicate with the security manager. The access request is either granted or denied by the security manager after it checks the security information on the service and device databases [4][5].

The first time that two devices attempt a connection, they need to be authenticated in order to form a trusted relationship called "pairing". Authentication is performed using a challenge-response methodology, based on the BD_ADDR and a link key. In older versions of Bluetooth (v2.0 and earlier), both devices use a common secret PIN code, between 4 and 16 characters, which is used for link key generation. Newer versions of Bluetooth (v2.1 and later) use the Secure Simple Pairing (SSP) for the pairing process. This protocol replaces the PIN with public key cryptography [5].

A piconet utilizes a unique frequency-hopping spread-spectrum (FHSS) technique, moving through 1600 frequencies per second to create a frequency hopping pattern. Each channel is used for only 625 microseconds before hopping to the next channel. Once connected, a slave will synchronize with the master's clock to get the correct frequency hopping pattern. It will be assigned a unique time slot for transmitting, which prevents collisions with other devices in the same piconet. In addition, since it hops over 79 frequency channels, the likelihood of interference with another piconet is low. Each hop is a time slot where data packets can be transferred. Since a packet can span up to 5 hops, the frequency will remain constant for that transfer. The master initiates regular transmissions to keep the piconet synchronized, while the slaves listen to the master time slots. The master sends transmissions on even numbered slots, and the slaves transmit on odd numbered slots. A slave will only transmit after it has received a transmission from the master [6].

The baseband layer establishes two kinds of links between the master and the slave(s): the synchronous connection-oriented link (SCO) and the asynchronous connection-less link (ACL). The SCO link has reserved slots at regular intervals for voice and data in a point-to-point link between master and slave. For any slots that are not reserved for the SCO links, the master can assign an ACL link to any slave on a per-slot basis, even if that slave already has an SCO link. The ACL links are used for data only, which can be transmitted in a point-to-multipoint manner to all slaves that are part of a piconet [7].

The Bluetooth protocol stack includes a variety of protocols. The Logical Link Control Adaption Protocol (L2CAP) passes data packets to the higher levels. The Link Management Protocol establishes and controls the link between the devices. The Radio Frequency Communications (RFCOMM) protocol provides serial communications that are widely used, and manage as many as 60 simultaneous active connections between two devices. The Service Discovery Protocol (SDP) allows devices to advertise the services that they offer and it also offers a way for them find one another [8].

III. Security Analysis

Bluetooth provides three basic security services:

- Authentication: verification of the identities of the

Bluetooth devices. User authentication is not provided.
- Confidentiality: ensuring that only authorized devices can access transmitted data.
- Authorization: allowing only authorized devices to access services.

Security services that are not included in the Bluetooth standard include audit, integrity and non-repudiation [9].

During authentication, one device is referred to as the claimant and the other device is referred to as the verifier. The claimant's role is to prove its identity and the verifier's role is to validate the claimant's identity. The claimant sends an access request to verifier, along with a secret 128 bit random number to be used as the link key. Then the claimant sends an authentication request along with its BD_ADDR. The verifier responds with a 128 bit random challenge number, AU_RAND. Then both devices perform authentication using the random number, the BD_ADDR and the link key. The claimant sends the resulting 32 bit signed response, SRES, to the verifier, which compares the response to its own computed value. When the values match, authentication is confirmed. If they do not match, authentication has failed.

Authorization is performed by checking the device database to determine if a device had previously been authorized as a trusted device. If it was, then access to services is granted. Otherwise trust must first be established before a device can be authorized.

Confidentiality is achieved through encryption using a stream cipher called E0. A keystream is generated using a link key and the BD_ADDR of the device, which is then combined with the plaintext to achieve the cyphertext [5].

IV. BLUETOOTH VULNERABILITIES

The security issues inherent in Bluetooth are largely due to the process of pairing one device to another. Various attacks can be performed before the pairing process has completed. Even after the devices are paired, an adversary can still gain enough information to be able to perform Man in the Middle attacks or to impersonate another device. [10]

The biggest factor in discussing Bluetooth vulnerabilities is the version of Bluetooth that is being used, and the security of communications between devices is only as strong as the weakest link, i.e. the device with the oldest (weakest) version. Since many older devices are still being used today, the vulnerabilities in the older versions of Bluetooth continue to be present. In the following we list and discuss some known Bluetooth vulnerabilities for various versions.

A. *Versions before Bluetooth 1.2*

Link keys are based on static unit keys and are reused in every pairing. Once it's revealed, a malicious device can use it to eavesdrop on the original device's connections. The malicious device can also spoof the original device or any device that is connected to it.

B. *Versions before Bluetooth 2.1 + EDR*

1. Short PIN codes are permitted, which can be easily guessed.
2. There is no PIN management which would be desirable at an enterprise level.
3. The keystream repeats after 23.3 hours of use, so if a connection lasts longer than that, an identical keystream will be used. This allows an adversary to decrypt messages.

C. *Versions 2.1 and 3.0*

1. Security Mode 4 devices can fall back to earlier versions when connecting to devices that don't support Security Mode 4. This includes Security Mode 1 which offers no security.
2. The use of static keys in SSP may lead to Man in the Middle attacks.

D. *Versions before Bluetooth 4.0*

1. The number of authentication challenge requests is unlimited. This allows an adversary to collect a large number of challenge responses which may reveal information about the secret link key.
2. The stream cipher E0 function is considered weak.

E. *All versions*

1. If link keys are stored improperly, then an adversary can view or even modify them.
2. The length of the encryption key can be as small as 1 byte.
3. There is no user authentication. Only device authentication is included in the Bluetooth standard.
4. A device can remain in discoverable/connectable mode indefinitely [11].

V. PERFORMANCE ANALYSIS AND RESULTS

For this paper, we performed a hands-on investigation of the security vulnerabilities associated with the Bluetooth Low Energy Protocol between two devices.

A. *Tools*

We installed the Ubertooth One tool, which is a fully open sourced hardware and software package that can be used for monitoring Bluetooth traffic. Ubertooth is an active eavesdropping device that can transmit and receive signals on the 2.4 GHz frequency. It can transmit and receive sensitivity comparable to a Class 1 Bluetooth device. The Ubertooth has several different transmitters and receivers. It was designed in KiCad, an open source electric design automation software. Michael Ossman presented Ubertooth at ShmooCon in 2011.

Figure 1 shows a diagram for the Ubertooth One device that was used in the experiments.

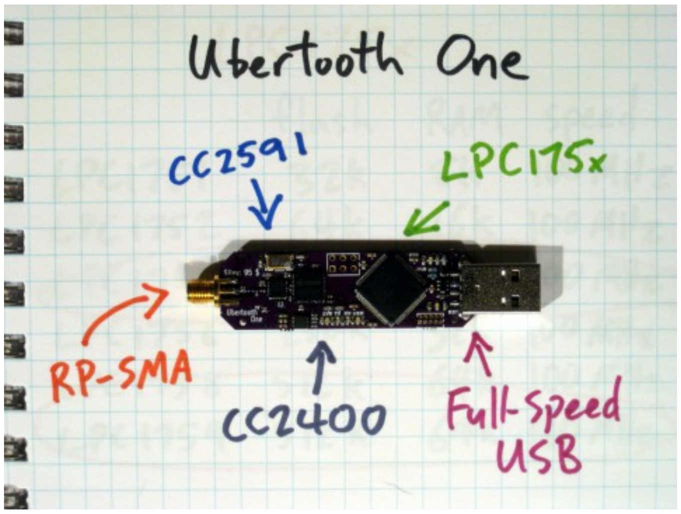

Figure 1: Ubertooth One Device

There are several tools in the Ubertooth suite that are available. One of them is a spectrum analyzer, which can be used to observe and analyze the traffic in the 2.4 GHz frequency range. Another tool is the Ubertooth scan, which can scan the channels in the 2.4 GHz Frequency. There is also a tool called HCItool which utilizes the Ubertooth to scan for devices associated with Bluetooth. This is great for the preliminary work in a penetration test of Bluetooth.

We utilized several other tools including Kismet and Wireshark to analyze the Bluetooth packets. Kismet is able to display the Bluetooth devices that are in range and capture packets. We used Wireshark because it has a Bluetooth plug-in that is able to show the raw packet data. We created a temporary file called ‘pipes’, and then instructed Ubertooth to capture packets and save them to this file. We were able to use Wireshark to view this file and observe a live feed of Bluetooth packets. We also included a program called Crackle in our tool suite, which is used to decrypt encrypted packets.

Installing all of the tools and getting them to successfully communicate with each other was challenging. This was partly due to the lack of documentation and instructions for implementing our test environment. Through trial and error, we were eventually able to get all of our tools working correctly and communicating, as we were able to observe the packets of our Bluetooth devices.

B. Initial Results

In all of our simulations, the first tool we used with the Ubertooth was the Spectrum Analyzer, which showed us the Bluetooth traffic and illustrated in Figure 2. This was followed by running the HCItool to scan for devices which were using Bluetooth.

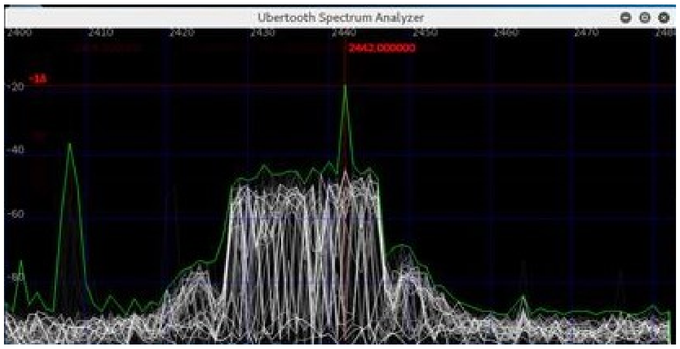

Figure 2: Spectrum Analyzer

As previously discussed, many of vulnerabilities in Bluetooth are exposed during the pairing process of one device with another. One of our main goals was to capture connection request packets and to view the encrypted packet communication from the Master to the Slave. However, we were not successful in achieving this. We were able to observe many packets, but not the desired connection request packets. After further investigation we determined that this was due to the Bluetooth devices which we were using. Many of the known Bluetooth vulnerabilities are prevalent in older Bluetooth devices, which utilize older, less secure versions of Bluetooth. For us, this was both a blessing and a curse. We were ‘blessed’ in one sense, because our personal devices did not have the Bluetooth vulnerabilities which we were attempting to exploit. However, for the purposes of our research this was also a curse, for the same reason.

Not to be defeated, we continued our quest and obtained additional devices in order to resolve this ‘curse’. But once again we were not successful in capturing the encrypted connection request packets. We believe that this was due to the Bluetooth security feature of frequency hopping, which we described earlier. The Bluetooth device is hopping frequency channels at a rate of 1600 hops per second. The Ubertooth is trying to capture all of the Bluetooth packets while also rotating through all of the channels. This is very hard to achieve so it ends up missing a lot of them.

C. A Degree of Success

We did find a degree of success with one of our devices. We were able to capture a CONNET_REQ packet from a Fit Bit device to an iPhone. The Fit Bit does not use encryption, since it is an older device and from our research we discovered that Fit Bit implemented encryption in the next model version. For the purposes of our test environment, the lack of encryption essentially rendered the use of Crackle moot. However, from the packets that we captured, we were able to see the Cyclic Redundancy Check, which is an error checking code commonly used with networks and storage devices.

Since we were not able to demonstrate the Crackle decryption process with any of our Bluetooth devices, we found some sample files that we were able to decrypt. Crackle uses vulnerability in the BLE pairing process that allows an attacker to guess the Temporary Key. From the data and the Temporary Key, the Short Term Key (STK) and the Long Term Key (LTK) can be collected. By possessing both the STK and LTK, all communication between the two devices can be decrypted. We were able to guess the Temporary key from the CONNECT_REQ packet that was sent between the master and slave.

VI. Bluetooth Risk Mitigation

As previously discussed, attacks against Bluetooth are due to the flaws that are found in the various versions of Bluetooth. These flaws allow an adversary to steal data, send messages, make phone calls and connect to the Internet, using the victim's device. When these types of vulnerabilities are discovered in a computer system, they can be resolved through application software patches. But for most Bluetooth devices, mitigating the vulnerabilities will necessitate that the firmware of the device be upgraded, which cannot be performed feasibly by the general public user community. The result is that many Bluetooth devices will remain vulnerable long after a mitigating solution becomes available [12].

Bluetooth uses wireless communication, and therefore the signal transmissions are susceptible to being captured by an attacker. The only way to ensure that Bluetooth communication between the two devices is secure is to pair the devices in a Faraday cage.

As with any wireless technology, there is no way to prevent all attacks and guarantee security. However, there are a number of countermeasures that can be achieved to provide reasonable security for Bluetooth communications. Some of those mitigation techniques are described below:

1. Provide an adequate level of understanding and awareness for Bluetooth users, to educate them in the proper security practices.
2. Change the default settings on devices to optimal standards.
3. Set the devices to the lowest power level necessary. This will ensure that devices remain within a secure range.
4. PIN codes should be long and random, to be more resistant to brute-force attacks.
5. Link keys should be based on combination keys rather than unit keys, to prevent Man in the Middle attacks.
6. Devices should be set to undiscoverable by default, except as needed for pairing, to prevent visibility to other Bluetooth devices.
7. Link encryption should be used for all data transmissions to prevent eavesdropping.
8. If using multi-hop communication, make sure that every link is encryption-enabled. Otherwise, the entire communication chain may be compromised.
9. Require mutual authentication for all accesses, to ensure that all devices on a network are legitimate.
10. Encrypt all broadcast transmissions to protect against them being intercepted.
11. Use the maximum encryption key size to protect against brute-force attacks.
12. Set a large minimum key size (preferably 128 bit) for all key negotiation processes, to protect against brute-force attacks.
13. Disable Bluetooth on devices when not in use. This minimizes exposure to threats.
14. Pairing of devices should only be performed as needed, and in a secure non-public setting where attackers won't be able intercept pairing messages. The pairing process is vital to Bluetooth security and users must be aware of the possibility of eavesdropping.
15. Security Mode 3 is highly recommended. It provides the highest level of security because it takes place at the link level, prior to link establishment.
16. Lost or stolen Bluetooth devices should be unpaired from any other devices for which is was paired. This will prevent an attacker from using the missing device to access any of the owner's other Bluetooth devices.
17. Users should never accept transmissions from unknown or suspicious devices. Only accept content from trusted devices [10].

VII. Discussion and Open Issues

Our initial research about Bluetooth vulnerabilities along with the availability of the Ubertooth tool made it feasible to sniff Bluetooth traffic and decrypt the packets, and our overall research paper was designed with that premise. Setting up all the Ubertooth applications on Kali and installing the Bluetooth plug-ins for Wireshark and Kismet were challenging. But after additional research, we were able to resolve those issues so that all of the applications were setup successfully.

Following the initial setup, we were able to observe a lot of Bluetooth traffic from Zigbee and other Low Energy applications. We tested it on a Bose speaker as well as a Motorola HX1 headset, but these devices did not work very well for what we were trying to show. There is limited security risk to have encryption setup on a Bose Speaker, since there no information stored on the speaker and there are no realistic concerns regarding eavesdropping of the Bluetooth speaker traffic. We tested the Motorola HX1 Headset but could not see any CONNECT_REQ packet sent from the phone and Headset. Further research revealed that it was not an LE device. We next tested the pairing between our smart phones and the Bluetooth system in a 2012 Honda Accord, and the result was the same. We had the best result with the Fit Bit device, with which we were able to get the CONNECT_REQ packets sent when the device successfully paired with the iPhone. We did not see any LL_START_ENC_REQ packets, which are displayed as SMP (Security Manager Protocol) packets in Wireshark and are the packets that carry the STK of the Master. We did not have any additional devices to test. In hindsight had we realized our predicament earlier, one possibility that we could have pursued would be to purchase a device that could have simulated a Bluetooth device with encryption enabled. There are several devices which can be programmed, such as the WICED Smart device.

Although we were not able to capture the files from our own devices, we still wanted to demonstrate the decryption process. We were able to find some sample PCAP files, which we used to show a Bluetooth attack. We used the program Crackle and were able to successfully decrypt the PCAP and find the LTK. Using the LTK you can decrypt other communication from the master to the slave.

This type of attack can happen in the real world. It may take some time, but by discretely setting up sniffers in many areas, an attacker could capture a lot of the traffic. Also, by using multiple Ubertooth devices an attacker would be able to capture many more packets on all the frequencies that Bluetooth uses. As already discussed, the chief concern with Bluetooth vulnerabilities is with the pairing process, so in most cases the devices would need to be pairing in order for the attacker to be successful. But any traffic that is unencrypted can be observed, and there are some older Bluetooth applications that do not use encryption. It is always recommended to use encryption, however, the use of strong security should not impact the system's performance [26, 27, 28, 32]. Hence, the use of lightweight cryptography can help to improve the performance [29, 30, 31].

It is also worth mentioning that Bluetooth are also vulnerable to attacks that cannot be prevented using cryptographic protocols, such as: jamming [18, 19], packet dropping [20, 21], wormholes [22, 23] and localization [24, 25].

VIII. CONCLUSION

Our main objectives for this paper were to study the Bluetooth technology and to perform a hands-on investigation of the various security vulnerabilities associated with Bluetooth. By utilizing the Ubertooth One tool in conjunction with Kismet, Wireshark and Crackle, we were able to perform spectrum analysis, packet sniffing and packet decoding. We analyzed our results and reported on the observed security risks associated with Bluetooth technology. We also discussed some mitigating techniques to resolve these issues.

Bluetooth technology has been widely available for over 15 years and it has gained worldwide acceptance. It has proven to be an easy and convenient solution for people that wish connect their various personal devices, for both voice and data transmissions. Yet the various Bluetooth versions that are in use today have a wide variety of security vulnerabilities. As the popularity of Bluetooth continues to grow and it is incorporated into more aspects of everyday life, it's very important that users understand the risks involved with using Bluetooth. Even more important is that they work to mitigate those risks by following the recommended security guidelines. Because Bluetooth operates in a short range, mobile, wireless network, there cannot be a fixed, central, trusted party. Therefore it is incumbent upon each user to be their own trusted party for securing their own Bluetooth communications.

REFERENCES

[1] Svenolof Karlsson and Anders Lugn, "The History of Bluetooth", excerpt from the book *Changing the world: The story of Lars Magnus Ericsson and his successors*.

[2] A. Davies, (2002, June). "An overview of Bluetooth Wireless Technology and some competing LAN standards", *Proc. Int. Conf. Circuits and Systems for Communications*, pp. 206-211, 2002, IEEE.

[3] P. Fowler, (2002, September). "5 GHz Goes the Distance for Home Networking," IEEE Microwave magazine, pp. 49-55.

[4] C. Bisikian, (2001, December). "An overview of the Bluetooth Wireless Technology", *IEEE Communications Magazine*, vol 39, pp 86-94.

[5] Zou, Yulong, Xianbin Wang, and Lajos Hanzo, (2015, May), "A survey on wireless security: technical challenges, recent advances and future trends." *arXiv preprint arXiv:1505.07919*.

[6] K.. Sairam, N. Gunasekaran and S. Rama Reddy, (2002, June). "Bluetooth in wireless communication", *IEEE Communications Magazine*, vol. 40, no. 6, pp. 90-96.

[7] R. Jordan and C.T. Abdallah, (2002, February). "Wireless communications and networking: An overview", *IEEE Antennas Propagation Magazine*, vol. 44, pp. 185-193.

[8] E. Ferro and F. Potorti, (2005, February). "Bluetooth and Wi-Fi wireless protocols: a survey and a comparison," Wireless Communications, IEEE, vol. 12, pp. 12-26.

[9] Lackner, Günther, (2013, November). "A Comparison of Security in Wireless Network Standards with a Focus on Bluetooth, WiFi and WiMAX." *International Journal of Network Security* 15.6: 420-436.

[10] Minar, Nateq Be-Nazir Ibn, and Mohammed Tarique, (2012, January). "Bluetooth security threats and solutions: a survey." *International Journal of Distributed and Parallel Systems* 3.1: 127.

[11] John Padgette, Karen Scarfone and Lily Chen, (Sept. 2008) (Rev. 1) (June 2012)."Guide to Bluetooth Security: Recommendations of the National Institute of Standards and Technology", NIST Special Publication 800-121.

[12] L. Carettoni, C. Merloni and S. Zanero, (2007, March). "Studying Bluetooth Malware Propagation: The BlueBag Project", *IEEE Security and Privacy,* vol. 5, no. 2, pp. 17-25.

[13] Shrivastava, Manish, (2012, December). "Analysis of Security Risks in Bluetooth." *International Journal of Computing Academic Research ($2305-9184)* 1.2: 88-95.

[14] Gomez, Carles, Joaquim Oller, and Josep Paradells, (2012, August). "Overview and Evaluation of Bluetooth Low Energy: An Emerging Low-Power Wireless Technology."*Sensors (Basel, Switzerland)* 12.9: 11734–11753.

[15] Mike Ryan, (2013, August). "Bluetooth: with low energy comes low security." In *Proceedings of the 7th USENIX conference on Offensive Technologies* (WOOT'13). USENIX Association, Berkeley, CA, USA, 4-4.

[16] C. T. Hager and S. F. Midkiff, (2003, March). "An analysis of Bluetooth security vulnerabilities," *Wireless Communications and Networking, 2003. WCNC 2003. 2003 IEEE*, New Orleans, LA, USA, 2003, pp. 1825-1831 vol.3.

[17] T. Hayajneh, G. Almashaqbeh, S. Ullah and A. Vasilakos, "A survey of wireless technologies coexistence in WBAN: analysis and open research issues," *Wireless Networks,* vol. 20, no. 8, pp. 2165-2199, 2014.

[18] Panyim, K.; Hayajneh, T.; Krishnamurthy, P.; Tipper, D. On limited-range strategic/random jamming attacks in wireless ad hoc networks. In Proceedings of the IEEE 34th Conference on Local Computer Networks, Zurich, Switzerland, 20–23 October 2009; pp. 922–929.

[19] Hayajneh, Thaier, Ghada Almashaqbeh, and Sana Ullah. "A Green Approach for Selfish Misbehavior Detection in 802.11-Based Wireless Networks." Mobile Networks and Applications 20.5 (2015): 623-635.

[20] Hayajneh, T.; Krishnamurthy, P.; Tipper, D.; Kim, T. Detecting malicious packet dropping in the presence of collisions and channel errors in wireless ad hoc networks. In Proceedings of the IEEE International Conference on Communications, Dresden, Germany, 14–18 June 2009; pp. 1–6.

[21] Hayajneh, T.; Almashaqbeh, G.; Ullah, S. A Green Approach for Selfish Misbehavior Detection in 802.11-Based Wireless Networks. Mobile Netw. Appl. 2015, 20, 623–635.

[22] Hayajneh, T.; Krishnamurthy, P.; Tipper, D.; Le, A. Secure neighborhood creation in wireless ad hoc networks using hop count discrepancies. Mobile Netw. Appl. 2012, 17, 415–430.

[23] Hayajneh, T.; Krishnamurthy, P.; Tipper, D. Deworm: A simple protocol to detect wormhole attacks in wireless ad hoc networks. In Proceedings of the IEEE 3rd International Conference on Network and System Security, Gold Coast, Australia, 19–21 October 2009; pp. 73–80.

[24] Hayajneh, T.; Doomun, R.; Krishnamurthy, P.; Tipper, D. Source—Destination obfuscation in wireless ad hoc networks. Secur. Commun. Netw. 2011, 4, 888–901.

[25] Doomun, R.; Hayajneh, T.; Krishnamurthy, P.; Tipper, D. Secloud: Source and destination seclusion using clouds for wireless ad hoc networks. In Proceedings of the IEEE Symposium on Computers and Communications, Sousse, Tunisia, 5–8 July 2009; pp. 361–367.

[26] Hayajneh, Thaier, Samer Khasawneh, Bassam Jamil, and Awni Itradat. "Analyzing the impact of security protocols on wireless LAN with multimedia applications." In Proc. 6th Int. Conf. SECURWARE, pp. 169-175. 2012.

[27] Hayajneh, Thaier, Bassam J. Mohd, Awni Itradat, and Ahmad Nahar Quttoum. "Performance and information security evaluation with firewalls." International Journal of Security and its Applications 7, no. 6 (2013): 355-72.

[28] T. Hayajneh, S Ullah, BJ Mohd, K. Balagani, "An Enhanced WLAN Security System with FPGA Implementation for Multimedia Applications," IEEE Systems Journal, 2015. (DOI: 10.1109/JSYST.2015.2424702)

[29] B. J. Mohd, T. Hayajneh, Z. AbuKhalaf, K. Yousef "Modeling and Optimization of the Lightweight HIGHT Block Cipher Design with FPGA Implementation," Security and Communication Networks, John Wiley, Vol. 9, No. 13, pp 2200-2216, 2016. (DOI: 10.1002/sec.1479)

[30] B. J. Mohd, T. Hayajneh, AV. Vasilakos, "A Survey on Lightweight Block Ciphers for Low-Resource Devices: Comparative Study and Open Issues," Journal of Network and Computer Applications, Elsevier, Vol. 58, pp 73-93, 2015. (DOI: 10.1016/j.jnca.2015.09.001)

[31] BJ Mohd, T. Hayajneh, M. Z. Shakir, K. A. Qaraqe, AV Vasilakos "Energy Model for Light-Weight Block Ciphers for WBAN Applications," In Proc. of IEEE 4th International Conference on Wireless Mobile Communication and Healthcare (IEEE MobiHealth'14), Athens, Greece, 2014.

[32] T. Hayajneh, R. Doomun, G. Al-Mashaqbeh, BJ Mohd "An energy-efficient and security aware route selection protocol for wireless sensor networks," Security and Communication Networks, John Wiley, Vol. 7, No. 11, pp 2015-2038, 2014. (DOI: 10.1002/sec.915)