Noname manuscript No.
(will be inserted by the editor)

BLE Injection-Free Attack: A Novel Attack On Bluetooth Low Energy Devices

Aellison C. T. Santos [1] · **José L. Soares Filho** [1] · **Ávilla Í. S. Silva** [1] · **Vivek Nigam** [2] · **Iguatemi E. Fonseca** [1]

Received: date / Accepted: date

Abstract Bluetooth Low Energy (BLE) is a variant of Bluetooth popular among Internet of Things (IoT) applications designed for devices with limited resources, which results in weak mechanisms of cryptography to create and exchange keys. Some serious attacks are based on forcing the key renegotiation of paired devices. Existing literature proposes the use of packet injection and even jamming devices to do so. This paper presents a new attack, called BLE Injection-Free attack, which exploits a novel technique to force the key renegotiation of devices. This technique exploits properties of the bonding list of devices and its defenses. The BLE Injection-Free attack enables Man-In-The-Middle and Denial of Service attacks to be carried out, depending on the BLE implementation. Our experimental results show that even when the key renegotiation cannot be forced, the functioning of the targeted device is still compromised.

Keywords IoT · BLE · Bluetooth · Security

1 Introduction

Bluetooth Low Energy is a variant of the Bluetooth protocol designed to accommodate the resource restrictions (energy and computational power) of devices, such as sensors. BLE optimizes communication in order to reduce the computational power and energy required by devices and still provide the same communication range as standard Bluetooth. BLE devices have been used in, for example, wearable IoT devices [3], industrial applications [1] and smart homes [2] for tasks such as automated home security and smart heating. It is expected that 48 billion Internet of Things (IoT) devices will be available in the market by 2021, and is predicted that nearly a third of those will be Bluetooth compliant [4].

Despite its success, the BLE technology suffers from severe security drawbacks as shown by [5][6][7]. While BLE devices make use of mechanisms for generating and exchanging long term keys, they have security vulnerabilities. If a malicious user sniffs packets exchanged during the initial process of pairing, then it is possible to infer the long term keys used by the devices. With those keys, the malicious user is capable of decrypting and peeking the data sent by the devices and even performing Man-in-the-Middle (MITM) attacks.

However, for devices that already performed the key exchange and used the bonding functionality without a malicious sniffing device being present, the attacker, even observing the connection process of the devices, cannot collect the needed information to infer the long term keys used by both parts of the connection. This happens due to the ability of storing the long term keys of the devices, which is what happens when the bonding functionality is used. The usage of bonding allows the users to skip part of the data exchange regarding the generation of the keys, which would give crucial information to the attacker, as both devices already have encryption keys stored. This functionality is a hindrance to the execution of the attack proposed by Mike Ryan [8], but it still can be defeated by using packet injection techniques.

If there is a failure in the packet injection process, the attacker can wait for the beginning of a new connection attempt, which reduces the chance for a successful

Aellison C. T. Santos
E-mail: aellisoncassimiro@cc.ci.ufpb.b

[1] Informatics Center – UFPB, Paraíba, Brazil
[2] Fortiss, Munich, Germany

attack as the attacker needs to be within the BLE range for a longer period of time (on the assumption that, in the current conditions, the attacker will manage to succeed in its attack). Still, such strategy can be optimized with the usage of jamming techniques to shut down the communication channels used by the devices, forcing the reconnection of the target BLE devices.

This work proposes a new BLE attack, called *BLE Injection-Free Attack*. The attacker is capable of compromising BLE devices without the need to inject a packet nor jamming for consecutive attempts to perform MITM attacks. Such objective is attained by exploring properties of the bonding list of vulnerable devices.

There are a number of advantages for the attacker in avoiding both packet injection and signal jamming that are taken in consideration in the conception of this attack. Firstly, due to time constraints, the physical space between the attacker and the target is a crucial property to attaining success on packet injection attempts. The distance also heavily influences the success rate of jamming attacks. Easily concealable jamming devices have a short range effectiveness as shown [9] of at most 1 meter. The use of more powerful jamming devices would disrupt many neighboring BLE devices which could trigger alerts and countermeasures. The BLE Injection-Free attack proposed here does not suffer from these limitations.

Finally, we demonstrate this attack experimentally on general purpose BLE devices available on the market. With a device to represent the the legitimate user, a device to represent the peripheral target and an attacker that can make use of up to 4 BLE interfaces, different approaches are used to handle the bonding requests and the bonding list, giving a broad view of the consequences of the attack according to the actions of the users.

When talking about countermeasures, packet injection can be defeated with the usage of more sophisticated cryptography techniques, larger keys or alphanumeric PINs [10], in Bluetooth's case. Still, the establishment of encrypted communication happens after the section of the packet exchange where the injection can be performed. Racing conditions also can be added or explored to hinder the attacker. Jamming can also be mitigated by using defenses such as frequency hopping. Indeed, BLE already has some kind of frequency hopping, which although predictable, can be improved to avoid signal jamming. In our BLE Injection-Free attack, such countermeasures are not effective as it does not use packet injection nor jamming, but follows correctly the BLE protocol.

Moreover, in many scenarios, such as in wearables, BLE devices are actually attached to users that may be moving. It is much harder to carry out packet injection and jamming attacks as the attacker would need to attach its devices to the user. In the proposed attack, as long as the attacker stays in range, the attack can be easily carried out without the user noticing anything abnormal.

After evaluating the related work in Section 2, the BLE technology is described and contextualized in Section 3, followed by an overview on the threats and security issues of BLE devices. On section 5 we tackle in details the proposed attack free of injection and jamming. In section 6, the advantages, severity and consequences of the attack are discussed while illustrating its use cases with concrete scenarios. It is also discussed how this attack can be mitigated. In Section 7, the experiments are described demonstrating the effectiveness of the attack in a BLE device. Finally, in Section 8, the presentation of the work is concluded and future work is discussed.

2 Related Work

On the research fields related to this work, some attacks stood out while building the comprehension of dangers and challenges related to the security of BLE mechanisms.

Firstly, we were greatly inspired by the work of Mike Ryan [8]. The idea of Ryan with his tool "Crackle" is to explore the vulnerabilities present in the pairing methods available on BLE to obtain the cryptographic key of a connection. As a consequence, the attacker can perform several attacks, from obtaining the transmitted private user data to Man-in-the-Middle attacks.

Ryan describes how the functionality of bonding can be used to complicate the attackers activity, adding the need for packet injection in one of the stages of the pairing process between the target and the legitimate user requiring a key renegotiation and, in some cases, jamming in order to perform the attack with success in scenarios where multiple attempts are needed. While defenses for the approach created by Ryan where implemented on Bluetooth version 4.2 and above, the adherence of the newer versions is happening in a slow pace.

Exploring properties of the algorithm responsible for the key exchange, Tomas Rosa complements Ryan's work arguing that, even when devices use stronger Diffie-Hellman key exchange methods, instead of the set of non-consolidated algorithms internal to BLE explored by Ryan, the attack still can be successfully performed

[11]. This happens due to how the messages are exchanged, where a device A sends a value C calculated over a temporary key, a random value r and two public parameters. The attacker can use the values already known and obtain by brute force a random value r', different from r, but still capable of calculating C and send it back to the device A.

With regards to the BLE technology in general, the research strands presented by Jasek [12] covering vulnerabilities on the technology were analyzed. In his work, Jasek raises 7 different basic approaches to attack BLE devices, which are: attacks on advertisements, passive interception, active interception, attacks on exposed services, attacks on pairing, whitelisting bypass and attacks related to privacy. The tool used to perform some of these attacks, titled Gattacker and created by Jasek, while having a smaller potential than Mike Ryan's Crackle, can also perform sniffing and Man-in-the-Middle attacks in simpler systems.

To search, identify and analyze possible targets to attacks, techniques such as War Driving can be used. War Driving defines the process of searching and scanning an area with a device controlled by a malicious user, which moves to extend its reach. Executing this method, the attacker usually uses a car (hence the technique's name) with devices attached capable of scanning and also probing close devices, looking for both basic information and straightforward vulnerabilities. During the DEF CON 24 conference, in 2016, Anthony Rose and Ben Ramsey demonstrated the dangers of war driving by reporting an experiment where, using a vehicle equipped with signal reception and communication devices, they looked for vulnerable BLE devices [13].

In [14], it is shown that some wearable applications disregard security measures to the point where neither the pairing or bonding are performed, hence the data is transmitted with little to no protection. As a consequence, it is possible to obtain data such as notifications of third party message applications (*e.g.*, Messenger, WhatsApp), allowing the malicious user to peek at the received messages of the targeted user.

Finally, on the topics of BLE security in general, it can be seen in the literature that the technology is still susceptible to packet sniffing, denial of service attacks, man-in-the-middle attacks and many other malicious procedures beyond the techniques already presented [7]. Furthermore, the dangers of the usage of BLE in IoT scenarios and the importance of its security is explored in [5][6][15]. Based on the available attacks in the literature, frameworks for testing and evaluating the security of devices were created in order to help developers to secure their applications [16].

3 Bluetooth Low Energy

Devices that use BLE play roles in multiple variants of the Master-Slave architecture. Messages are exchanged over 40 frequency channels where 3 are dedicated to advertisement of BLE devices, informing other devices of their presence and providing information about its capabilities. Using the information given by the advertisement messages, other devices can initiate connections performing the pairing process. Once two BLE devices are paired, they exchange messages using the 37 remaining communication channels on the Bluetooth frequency spectrum. Also, periodically, the channel used for the communication changes (channel hopping).

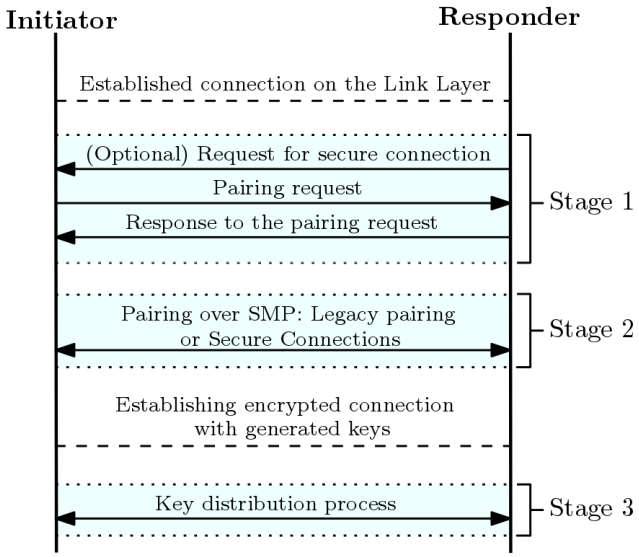


Fig. 1 Message exchange model of the Bluetooth Low Energy protocol of the pairing procedure. Adapted from [17].

After recognizing a connection request from the initiator, the pairing process develops into three stages, as shown in Figure 1.

At the first stage, the pairing mechanism to be used is chosen. At the second stage, depending on the chosen mechanism, the necessary data to generate the cryptographic key is exchanged, and in the third stage the keys are exchanged.

Depending on the pairing method, a Personal Identification (PIN) number is used as one of the inputs for the key generation.

Up to the version 4.2 of BLE, there are three pairing methods available.

- **Just Works**: Uses 000000 as the PIN in order to generate the key. Commonly used in devices without a communication interface.

Table 1 New Bluetooth features according to its version.

Bluetooth Version	New Key Features
Bluetooth 4.0	Bluetooth Smart (Low Energy)
Bluetooth 4.2	Low Power IP LE Privacy 1.2 LE Secure Connections
Bluetooth 5.0	Slot Availability Mask (SAM) LE Long Range LE Advertising Extensions

– **Pass Key Entry**: Uses a value between 0 and 999,999 as the PIN to generate the key. Used by devices with more sophisticated resources for communication.
– **Out of Band (OoB)**: Can be used when a device has means of communication external to BLE (e.g. Wi-Fi, NFC).

Even using other methods to exchange the keys in a safe way, capable of defending against attacks presented by Ryan using the Diffie-Hellman key exchange algorithm, for example, the methods of connection still have been proven unsafe. Tomas Rosa argues and demonstrates that Ryan's work can be extended to defeat such methodologies by using packet injection [11].

It is worth pointing out that throughout its development, aside from continuously improving in range and speed, there were many features that gradually changed how the Bluetooth technology was perceived by its public. The most relevant of the latest versions and its changes can be seen in Table 1. The introduction to the BLE, as discussed, brought attention to the security perspective of the technology, and attempts to correct security issues were made in version 4.2. On version 5, while great improvements to the capability of the devices were made, there were no changes in its security mechanisms.

4 Vulnerabilities

The Bluetooth technology possesses many vulnerabilities in various stages of the communication process of its devices, as it can be seen in the work of the U.S. National Institute of Standards and Technology (NIST) [18]. Aside from proper attacks further explored in [16], security issues and vulnerabilities on different versions of the Bluetooth technology are discussed in [18]. On Table 2, the issues and vulnerabilities that affect Bluetooth 4.2 can be seen.

Even though most of the attacks discussed here are similar or based on the threats presented in Table 2, they can go beyond what was already classified by NIST,

Table 2 Security issues and vulnerabilities on Bluetooth 4.2 recognized by NIST.

Vulnerabilities and security issues	NIST Description
Just Works does not provide MITM protection on pairing	Unauthenticated link keys generated using Just Works pairing are accepted on SSP.
Static or weak SSP EDCH key pairs	ECDH keys, used to enforce SSP eavesdropping protection, can be weak or the same one can be used for every SSP process.
Backwards compatibility on Security Mode 4	The Security Mode used for a connection can be established based on an outdated version supported by one of the devices.
Repeatable authentication attempts	Bluetooth allows an unlimited amount of authentication requests from a user. There is no interval for authentication challenge requests.
LE privacy by address	Low energy privacy may be compromised if a device's address is captured. There is no address privacy mechanism for Bluetooth 4.2.
LE Security Mode 1 Level 1 offers no protection	LE devices can perform connections without using protection mechanisms for its pairing.
Link keys can be stored improperly	Link keys can be manipulated by an attacker if they are not securely stored or handled.
Unknown strength of pseudo-random number generators	Bluetooth implementations allows Random Number Generators to produce static or periodic numbers.
No user authentication	There is no application-level security. Only device authentication is provided by Bluetooth.
End-to-end security is not enforced	Only links are encrypted and authenticated. Instead of protecting the integrity of the data from a point to another, data can be decrypted at intermediate points.
Limited security services	Known and simple security services are not part of the Bluetooth standard. They must be provided by applications' developers.
Discoverable and connectable devices	Devices should stay in a discoverable/connectable mode for a limited amount of time. This is not enforced by Bluetooth.

applying these malicious techniques with different perspectives.

For instance, with regards to advertisements, attackers can personify a target devices by cloning its basic information (name, services provided, address) to fool legitimate users. Also, attackers can use jamming signals to disable the advertisement channels, preventing connections from starting [12][9]. War driving methods can be performed in order to search and scan possible target devices [13][19], helping an attacker to map an area and evaluate possible attacks working only, but not limited to, on the scope of advertisement messages. Similarly, an attacker that can identify and communicate with a vulnerable device can make use of its protocols to perform attacks. Relay attacks can be used as an example of such, presented by Ho et al. [20]

Briefly, these attacks have been discussed on section 2. For this work, a focus is given to vulnerabilities related to pairing mechanisms available on BLE. In all mechanisms, long term keys are generated using a PIN value and data exchanged during the pairing process. In the methods Just Works and Pass Key Entry, widely used by BLE devices, all the information needed to generate the key, except for the PIN, are sent as non-encrypted text, which means that these can be obtained by devices sharing the middle.

While the PIN value is not explicitly exchanged during the communication, the attacker can infer it. This happens due to the simplicity of the PIN, which in the Just Works method is always 000000 and in the Pass Key Entry method it's a number between 0 and 999,999. Such value can be easily guessed using brute forced. The pairing method OoB is the one capable of providing the most security properties when correctly used, but as stated by Ryan and Jasek, the amount of devices available in the market using OoB as its pairing mechanism is scarce.

With the argumentation in Section 2, given that a attacker is listening to the middle and having access to the data being transmitted during the pairing process, the long term keys can be inferred and the target's messages can be decrypted using Mike Ryan's proposed tool. In case of pairing with bonding, packet injection and jamming can be used overcome this defense mechanism. Still, both techniques have their own downsides, as shown bellow.

4.1 Limitation of Packet Injection attacks

At first, attacks that use packet injection may seem advantageous, as many of the classic defense mechanisms against such attacks require computational power beyond what is available for the devices in IoT scenarios [21]. The open nature of the communication middle in such scenario is also another strong point in favor of approaches that use packet spoofing. Indeed, BLE devices fit well on the category of devices vulnerable to exploits based on packet injection, but a few other properties derived from the Bluetooth application scenarios also weight in the possibilities of attaining success in an attack.

Due to the usage of energy-efficient transmitters, BLE devices usually communicate within a range of 10 meters without barriers. To perform an injection attack in scenarios where the range of connection is this short can prove to be a threat to the attacker, as the legitimate user most likely will be present on the scenario. The spoofing devices must be concealed, thus generating requirements regarding limitation of size of the device, if the attacker has no other means of concealment of the equipment used.

Furthermore, the packet processing capacity of the attacker device and its proximity to the target can be the source of race condition problems, putting the malicious user in unfavorable circumstances. In case the attacker finds himself in a disadvantageous position to the attack, he can still try to predict the correct moment to spoof the packets in order to beat the legitimate user, but this will only worsen the odds of the attacker.

Finally, if the objective of the malicious user is to obtain the data transmitted by the devices, no special hardware is needed since usually there is no problem in waiting a bit longer to decrypt the messages, once you have the devices LTKs. This is counterintuitous to the need of high performance hardware for packet spoofing capable of handling situations with race conditions.

4.2 Limitations of Jamming Attacks

Signals used to perform jamming attacks can be broadcasted by small and considerably cheap devices, as shown by the work of Brauer et al. [9], excellent for attacks in IoT scenarios[1]. However, they are only effective when placed close to the target (at least one meter from it). This means that the attacker needs to have physical access to his targets. As some targets are placed in closed rooms, such as in smart homes and factories, the attacker has to first overcome the security barriers of such rooms, that is, gain access, place the jamming device near the target, and leave the room without raising suspicion.

[1] Devices that broadcast jamming signal capable of affecting larger areas are both costly (in regards to production cost and energy consumption) and easier to be detected by the legitimate user.

Moreover, due to the range limitation of jamming devices, once placed in its position for the attack, the targets also become fixed. In order to change the target of the attack, a new disruptive signal emitter must be placed next to new new target or the previously placed emitter must be relocated and placed in reach of this new target. The jamming device also cannot be moved or detected by users. For instance, if the attacker places the device on the floor, it might be swept away while the room is being cleaned. Thus it is important that the attacker finds a suitable location for the jamming device. This might not be necessarily easy to find.

Properties regarding the proximity of the devices also imply in a lower reliability of the attack should the target device move. This is specially a problem for scenarios with wearable devices. The attacker can attempt to append the device to the user or keep its device in range, which might prove to be a challenge. One example of such scenario is an extension of what is shown in [22], where the security of smart wristbands is discussed and worrying results are presented.

There are few approaches that can be used to mitigate jamming attacks. Some of them are discussed in [23]. While we could not find any countermeasure effectively used by BLE devices against jamming, it can be noted that adapting some of these measures to the BLE technology may come at the cost of a remodeling of the channel hopping technique currently used. Another method uses multipe coordinated devices close to each other, making possible to detect possible ongoing jamming attacks. Similarly, when a BLE device believes that its channels are being disrupted, it can follow a alternatively communication protocol using a technology external to Bluetooth (Out of Band).

5 BLE Injection-Free Attack

Having in mind that the attack proposed by Ryan consists of packet injection and the negative points of using this approach, it was possible to develop a new method capable of deleting an instance of the bonding information list of a device. By doing so, the attacker has no more need to inject packets requesting the LTK renegotiation, as the key will be deleted in this process. Since there will be no more need to spoof packets in a specific space of time, there will be no need for jamming in case of failure to inject.

Threat Model To perform the BLE Injection-Free attack, the attacker must possess a device with multiple BLE interfaces – enough to fill the target's bonding list – or, if only one interface is available, be capable of virtually change its address. The attacker can place its device anywhere inside the communication range of the target, as distance has no influence on the fulfillment of the attack. If performing the injection-free method on a moving device, the malicious user only has to maintain its device in communication range of the target during the execution of the attack.

While the proposed attack can be applied to any device with bonding capabilities, it is not recommended if the attacker needs to have access to physical interfaces of the targeted peripheral in order to finish the pairing process. Still, the attack can be feasible in such scenarios, specially if access to the device is open to the public (*e.g.*, in showcase) or if it can be accessed while left alone by the legitimate user.

The key idea of the BLE Injection-Free attack is to exploit the fact that BLE devices can only store at most a (small) number of keys. Once the maximum number of keys have been installed and a new bonding request is received, the device forgets one of the installed keys, k, in order to give place for a new key, k', for the received bonding request. This causes, however, the device paired using k to no longer be able to communicate with host. It will need to negotiate a new key with the peripheral. Thus, if the attacker can fill the bonding list of the devices, the BLE target deletes all existing keys exchanged with legitimate devices. This will force all legitimate devices to reinitiate the bonding procedure without the need of packet injection. Other implementations handle bonding requests differently when the list is full. For example, instead of forgetting keys, they deny connection to new users or simply allow unsecure connections without bonding. Such strategies for when the list of keys is full have their own security problems, *e.g.*, subject to Denial of Service attacks.

Once the connection between the master and slave is established using bonding, both devices store the information needed to skip pairing stages on the next connection. This information is kept on a list of limited size. Moreover, slave devices usually have limited interfaces, becoming incapable of displaying to the user which devices are currently stored in the bonding list, making them easier targets for this injection-free attack.

In Figure 2, it is possible to illustrate the attack procedure. In it, we assume that the slave device tagged as "Responder" is the target and possesses a bonding list of size 3. The "Initiator", representing the legitimate device, makes a connection and establishes the keys. After this, the bonding information is now stored in the bonding list of both devices, which means that there will be no need to calculate new LTKs in the next connection attempt. The attacker then proceeds to es-

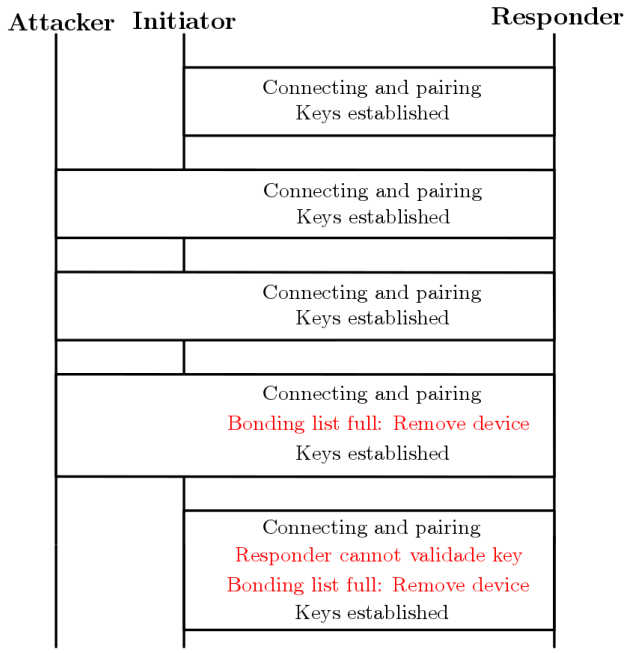


Fig. 2 Diagram modelling an example of the injection-free attack.

tablish a sequence of connections with the target device, using multiple BLE interfaces or simply spoofing other MAC addresses. After the third connection, the bonding list will have reached its maximum capacity, and the next connection attempt will be handled according to the device implementation. There are three ways to handle a bonding request with a full bonding list:

- **Data deletion**: The device can clear one of the slots in its bonding list, erasing the information needed to connect with a device without calculating a LKT. The device which is currently attempting to establish a connection will now take the slot of the erased device (Case presented in Figure 2).
- **Pairing without bonding**: The device keeps using the bonding functionality for every device that already is in the list. Connection attempts from non-bonded devices can be accepted by the slave device, but the bonding functionality will not be used and for every connection new LTKs will be calculated.
- **Denial of service**: The device keeps using the bonding functionality for every device that is already in the list. Connection attempts from non-bonded devices will be ignored.

6 Discussion

With the knowledge presented in the previous section, it can be noted that in the three possible ways to handle the attack (as long as no extra resources are used, as it will be discussed shortly) the functionalities of the device will either be compromised or it will become susceptible to attacks discussed on Section 2.

The following subsections argue about possible scenarios where the attack can be performed and counter measures.

6.1 Scenario 1: Data deletion

In this scenario, the objective of the attacker is to remove the information of legitimate devices from the bonding list of the target. By accomplishing his goal, the attacker can proceed with the executions of the techniques presented by Mike Ryand and decrypt data from sniffed packets. The attacker can further extend his actions in order to perform a Man-in-the-Middle attack.

Intuitively, it can be seen that this strand of the attack can eliminate the privacy guarantee of a device and its users.

This strategy can be applied on devices with bonding capability that transfer data of interest for the attacker, be it for the sake of obtaining it or to change the desired values in a more sophisticated attack. Use cases for this scenario includes image, audio or text transfer. One instance of such are scenarios with health monitoring equipment (e. g. blood pressure sensor, heart rate sensor).

6.2 Scenario 2: Pairing without bonding

In this scenario, the objective of the attacker is similar to the discussed in the previous subtopic, but the main difference is that some devices' information is already stored in the bonding list. These devices can perform connections using their pre-established LTKs, while new devices cannot. Each new device that attempts to perform a connection will have to go through the key generation and exchange process. This happens to every attempt of every device not bonded to the target.

Having this in mind, the set of targets of the attack when the affected devices perform pairing without bonding is the same of the previously presented attack strand.

Thus, an attacker can, for instance, with aid of techniques such as wardriving, attempt to fill bonding list of devices that have not reached all its final users (attacking factories or stores), proliferating compromised devices in the market.

6.3 Scenario 3: Denial of Service

A counter argument that can be used regarding the effectiveness of this attack is based on the Denial of Service approach, where every device that has not been added to the bonding list once it's full will have its service denied by the target. The legitimate devices that are already bonded will still be able to perform connections with the target device, it must be emphasized that limiting the amount of devices to which the peripheral can communicate to goes against the idea of using bonding lists. In other words, after performing the attack, the legitimate users lose their capability of connecting the targeted equipment to new devices, making it unusable in multiple scenarios.

To make this clear, a scenario where a smart lock is used to block the access to a factory's warehouse can be analyzed. The warehouse holds vital resources to keep the factory running, so periodically resources must be stored and withdrawn from there. Ideally, giving the importance of the material in the warehouse, a select set of employees will be able to unlock the gates leading to the resources, and, by using the bonding functionality on the lock, if a attacker performs the injection-free attack and the lock does not deletes data from the bonding list to avoid exploitation over the key renegotiation, no other employee will be able to become a legitimate user of the peripheral. As a consequence, the attacker reduces the number of employees capable of accessing the device, requiring that those with this ability stay close to the environment of the attack in order to unlock the gates.

Analogous are the scenarios with BLE household locks (smart bolt locks and smart door locks) for home safety, where the number of users capable of accessing a house can be even more limited than the number of its inhabitants.

6.4 Possible Countermeasures

Having in mind the implementation options of the developers of the software use by the BLE device, there is a certain level of freedom for the development team to choose how the bonding list and bonding requests are handled. The simple option of deleting data from the list can be used as an example of how a device can mitigate this attack to avoid a denial of service.

Nonetheless, if done manually by the user, resources must be directed to the development of communication interfaces to enable users to choose when the list should be cleared and to identify devices in the list. If done automatically, the deletion criteria must be established carefully as not to remove a legitimate device from the list and make make it vulnerable against packet injection attacks.

In contrast, considering a scenario where the legitimate user intends to have a fixed amount of devices bonded to the peripheral, the developers can adopt a strategy of denial of service to new devices, ignoring key renegotiation request to avoid address spoofing. It is important to emphasize that, in order to this defense approach to work, it is necessary to assure that all legitimate devices are bonded to the target previously to the accomplishment of the attack.

On a less computationally expensive approach, the legitimate user of the peripheral can seek to ensure that there are no other devices close to a potential target of attacks, lessening the odds of the accomplishment of packet injection methods and requiring the usage of stronger jamming equipment by the attacker. Such strategy can be successfully used in scenarios where the functioning of a peripheral is critic and the user has ways to evaluate the communication middle in order to search for attacks. On a environment where the user has complete control over the presence of BLE devices, which is a rare scenario on day-to-day applications, the data deletion approach for handling the bonding list can be used without problems.

Another sophisticated approach for a scenario with plenty of devices make use of IoT concepts in its deployment. By means of advertisement packets, BLE devices can attempt to send data to close devices reporting the state of the communication middle. In case the device suddenly stops advertising or, by analyzing the data being transmitted, detects and reports an anomaly in the network, a master device can be notified in order to activate defenses against jamming and other attacks. Such scenario requires great efforts by the development team, since a protocol for such organized strategy must be developed and, in its turn, it would probably only be respected by devices interacting with equipment developed by the same company due to different standards in the market.

Another approach independent of the usage of multiple interacting devices can use a pseudo-random information deletion tactic to create free slots in the bonding list. This strategy is similar to what is used in [24] to defend servers against distributed denial of service attacks, where a malicious user is attempting to take all the resources of a server. This approach is also used to defend VoIP application servers [25]. As a defense mechanism, devices are removed from the bonding list, at first, randomly in order to free slots in the list. As the referenced work for this approach suggests, some statistics can be used to evaluate the probability of a

device being legitimate or not, making the approach even more efficient.

One more approach to hinder the attacker's capacity of spoofing devices is presented in [26], where a device authentication method for BLE is proposed. Together with this approach, a white list could be created to ensure that the devices that are allowed to connect belong to legitimate users. Still, the creation and handling of such a list might be subjected to its own vulnerabilities.

7 Attack Demonstration

To demonstrate this attack and evaluate the security of BLE devices, the injection-free attack was tested at the Network Laboratory (LAR) at the Federal University of Paraíba (UFPB), using as a target the BLE Pioneer Baseboard programmable board with the CY8CKIT-142 BLE module, produced by Cypress. This board is available in the market, applicable on real-world scenarios, and is multipurpose, giving its user the capacity to implement its functions according to its application scenario. It was possible to observe messages printed by the board using the Tera Term software as a visual interface, connected to the board using a USB connection. With this hardware, it was possible to program the connection and bonding functionalities using the BLE technology and evaluate the attack in real time. The programmed device place up to 4 devices in its bonding list.

Fig. 3 Hardware used to perform a BLE Injection-free attack.

A computer connected to three BLE interfaces and a smartphone were used as malicious devices of the attacker. The board was used as the legitimate peripheral and the smartphone as the legitimate user's device. The equipment is displayed in Figure 3. It is important to emphasize that attackers can run the BLE Injection-free attack using a single smartphone, for instance, if it can be configured to continuously spoof different addresses per connection.

All the experiments where performed using Bluetooth version 4.2, given its current strong presence in the market. No experimentation regarding bluetooth 5.0 was made, but given that the main changes introduced in the new version are about performance, it is assumed that this version is also vulnerable to the BLE injection-free attack.

7.1 Scenario Composition

For the attacks presented here, the legitimate user uses the Cysmart application, available on app markets for Android and iOS devices, installed in a Android smartphone to connect to the board. By its turn, the attacker possesses 4 BLE interfaces capable of performing connections with the target. This layout of devices for the attacker and legitimate user can be seen on Figure 4.

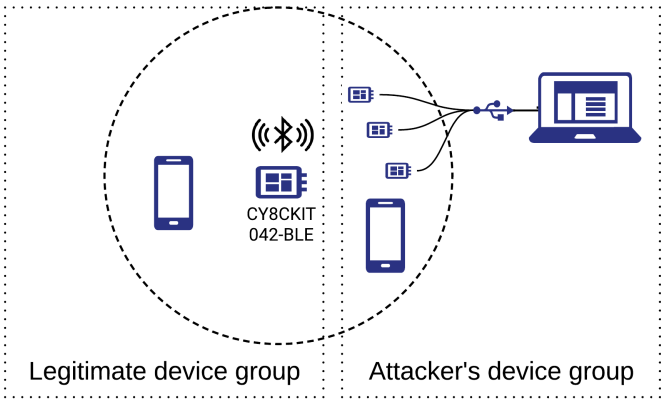


Fig. 4 Layout of devices used in the experiment.

As shown in the Figure 4, the only prerequisite to perform this attack on the proposed scenario is that the board is discoverable and in range of the connecting device.

The methodology for the experiment is the following: The legitimate user's device connects with the board, being the first device to be added to the bonding list. After this, the injection-free attack is performed, overflowing the bonding list. At the end, one device will attempt to connect with the board depending on the approach used to handle requests when the list is full, as it will be described.

7.2 No Bonding Request Handling Approach

For the first example, no handling approach was implemented to deal with bonding requests and a full bonding list, leaving the device to manage the situation according to its standard event management process.

The results of this experiment can be seen on Figure 7.2, where the output returned by the board shows the information about its current state. When the board returns to an advertisement state, it prints how many devices it has already bonded and the device addresses as they are stored. It can also be noted, comparing the Figure 7.2 (a) and Figure 7.2 (b), how the data structure of the list works. Every time a device is added to the list, its information is placed at the beginning of the structure.

In Figure 7.2 (c), we can see a connection request of a legitimate user after the attacker fills up the bonding list. The connection was denied, characterizing a denial of service for every new device attempting to perform a connection with the board. The devices that are still bonded with the board can perform connections without any problem.

7.3 Data Deletion Approach

Having in mind how devices are added to the list, as shown in Figure 7.2, a removal system was implemented to handle requests when the bonding list is full. The removal of devices works under a FIFO approach, meaning that if an attacker wants to remove the first device to bond with the board, n connections will have to be performed with the target, where n is the size of its bonding list.

After reformulating the handling of bonding requests with such approach, the results presented in Figure 7.3 were obtained. Figure 7.3 (a) shows the filled bonding list, while Figure 7.3 (b) shows the bonding list after a new device is added, removing the first device to bond with the target. When trying to reconnect to the board, the removed legitimate device sends the needed information to request a connection with the previously established LTKs, but the peripheral has no more reference for this device. Due to this, an authentication error is shown, as seen in Figure 7.3 (c).

The actions following this connection attempt will vary according to the implementation of the legitimate device. In the scenario presented here, the connection is terminated, an authentication error message is shown to the user and the board restarts the advertisement process once again. Still, it is perfectly possible that the application used by the legitimate device attempts to restart the connection, making the device undergo another pairing and bonding process.

8 Conclusion and Future Work

On the current trend usage of Internet of Things, applications such as Industry 4.0, Smart cities and ubiquitous technologies will be strongly linked to society's day-to-day life. BLE technologies are being heavily used and due to their limited resources, they are subject to cyber-attacks. Here, we propose a new attack, called BLE Injection-Free attack, which in contrast to existing attacks, does not require the attacker to inject packets nor jam the communication of BLE devices in order to require a LTK renegotiation.

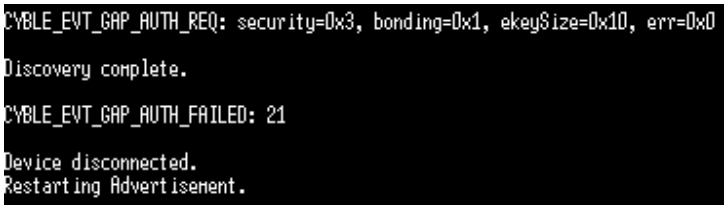
As future work, some of the properties of Bluetooth Low Energy and the automation of the BLE Injection-Free attack are being studied, as well as the implementation of defense mechanisms and conception of other countermeasures for this attack.

References

1. Manik Grover and Suraj Kumar Pardeshi and Nirbhow-Jap Singh, Bluetooth low energy for industrial automation, 2nd International Conference on Electronics and Communication Systems (ICECS), (2015)
2. Developex, BLE in smart home devices, http://developex.com/blog/ble-in-smart-home-devices/, accessed in 29/03/2019 (2017)
3. Richa Dham and Pushek Madaan, The role of Bluetooth Low Energy in wearable IoT designs, https://www.embedded.com/design/connectivity/4437074/The-role-of-Bluetooth-Low-Energy-in-wearable-IoT-designs, accessed in 29/03/2019 (2014)
4. Khatod Varsha Ritesh and Agata Manalova and Maria Nenova, Abridgment of bluetooth low energy (BLE) standard and its numerous susceptibilities for Internet of Things and its applications, 2017 IEEE International Conference on Microwaves, Antennas, Communications and Electronic Systems (COMCAS), p. 1-5 (2017)
5. Angela Lonzetta and Peter Cope and Joseph Campbell and Bassam Mohd and Thaier Hayajneh, Security vulnerabilities in Bluetooth technology as used in IoT, Journal of Sensor and Actuator Networks, 7(3):28 (2018)
6. Daniel Celebucki and Maj Alan Lin and Scott Graham, A security evaluation of popular internet of things protocols for manufacturers, 2018 IEEE International Conference on Consumer Electronics (ICCE), p. 1-6 (2018)
7. Harry O'Sullivan, Security Vulnerabilities of Bluetooth Low Energy Technology (BLE), Tufts University, (2015)
8. Mike Ryan, Bluetooth: With Low Energy Comes Low Security, 7th USENIX Workshop on Offensive Technologies, (13): 4–4 (2013)
9. S. Brauer and A. Zubow and S. Zehl and M. Roshandel and S. Mashhadi-Sohi, On practical selective jamming of Bluetooth Low Energy advertising, 2016 IEEE Conference on Standards for Communications and Networking (CSCN), 1–6 (2016)

(a) First device connected

(b) Bonding list full



(c) Connection attempt after completely filling the list

Fig. 5 Injection-free attack on a device with bonding list of size 4 without full list handling.

(a) Full bonding list

(b) List after replacing a device

(c) Connection attempt from the removed device

Fig. 6 Injection-free attack on a device with bonding list of size 4. When receiving a bonding request with the list full, the peripheral removes the oldest device from the list.

10. Bradley Reaves and Thomas Morris Analysis and mitigation of vulnerabilities in short-range wireless communications for industrial control systems, International Journal of Critical Infrastructure Protection, 5(3-4):154–174 (2012)
11. Tomas Rosa, Bypassing Passkey Authentication in Bluetooth Low Energy, IACR Cryptology ePrint Archive, 309 (2013)
12. Sławomir Jasek, Gattacking Bluetooth smart devices, Black Hat USA Conference, (2016)
13. Anthony Rose and Ben Ramsey, Picking Bluetooth Low Energy Locks from a Quarter Mile Away, DEF CON 24, https://www.youtube.com/watch?v=8h9nbMB1eTE, accessed in 29/03/2019 (2016)
14. Matteo Langone and Roberto Setola and Javier Lopez, Cybersecurity of wearable devices: an experimental analysis and a vulnerability assessment method, 2017 IEEE 41st Annual Computer Software and Applications Conference (COMPSAC), 304-309 (2017)
15. Yanzhen Qu and Philip Chan, Assessing vulnerabilities in Bluetooth low energy (BLE) wireless network based IoT systems, 2016 IEEE 2nd International Conference on Big Data Security on Cloud (BigDataSecurity), IEEE International Conference on High Performance and Smart Computing (HPSC), and IEEE International Conference on Intelligent Data and Security (IDS), 42-48(2016)
16. Apala Ray and Vipin Raj and Manuel Oriol and Aurelien Monot and Sebastian Obermeier, Bluetooth Low Energy Devices Security Testing Framework, 2018 IEEE 11th International Conference on Software Testing, Verification and Validation (ICST), 384-393 (2018)
17. Kai Ren, Bluetooth Pairing Part 1 Pairing Feature Exchange, https://blog.bluetooth.com/bluetooth-pairing-part-1-pairing-feature-exchange, accessed in 29/03/2019 (2016)
18. John Padgette and John Bahr and Mayank Batra and Marcel Holtmann and Rhonda Smithbey and Lily Chen and Karen Scarfone, Guide to bluetooth security, NIST Special

Publication, SP 800-121 Rev. 2, (2008)
19. Whitehouse, Ollie and others, War nibbling: Bluetooth insecurity, white paper @ stake Inc., (2003)
20. Grant Ho and Derek Leung and Pratyush Mishra and Ashkan Hosseini and Dawn Song and David Wagner, Smart locks: Lessons for securing commodity internet of things devices, Proceedings of the 11th ACM on Asia Conference on Computer and Communications Security, 461–472 (2016)
21. Yingying Chen and Wade Trappe and Richard P. Martin, Detecting and localizing wireless spoofing attacks, Sensor, (SECON'07) 4th Annual IEEE Communications Society Conference on Mesh and Ad Hoc Communications and Networks, 193–202 (2007)
22. Qiaoyang Zhang and Zhiyao Liang, Security analysis of bluetooth low energy based smart wristbands, 2017 2nd International Conference on Frontiers of Sensors Technologies (ICFST), p. 421-425 (2017)
23. Kanika Grover and Alvin Lim and Qing Yang, Jamming and anti-jamming techniques in wireless networks: A survey, International Journal of Ad Hoc and Ubiquitous Computing, 17(4):197–215 (2014)
24. Yuri Gil Dantas and Vivek Nigam and Iguatemi E. Fonseca, A Selective Defense for Application Layer DDoS Attacks, JISIC, 75–82 (2014)
25. Marcilio Lemos and Yuri Gil Dantas and Iguatemi Fonseca, and Vivek Nigam and Gustavo Sampaio, A selective defense for mitigating coordinated call attacks, 34th Brazilian Symposium on Computer Networks and Distributed Systems (SBRC), (2016)
26. Tianbo Gu and Prasant Mohapatra, BF-IoT: Securing the IoT Networks via Fingerprinting-Based Device Authentication, 2018 IEEE 15th International Conference on Mobile Ad Hoc and Sensor Systems (MASS), 254-262 (2018)