

Securing bluetooth low energy locks from unauthorizedaccess and surveillance

Anthony Rose, Jason Bindewald, Benjamin Ramsey, Mason Rice, Barry Mullins

▶ **To cite this version:**

Anthony Rose, Jason Bindewald, Benjamin Ramsey, Mason Rice, Barry Mullins. Securing bluetooth low energy locks from unauthorizedaccess and surveillance. 11th International Conference on Critical Infrastructure Protection (ICCIP), Mar 2017, Arlington, VA, United States. pp.319-338, 10.1007/978-3-319-70395-4_16 . hal-01819142

HAL Id: hal-01819142
https://inria.hal.science/hal-01819142

Submitted on 20 Jun 2018

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.



Distributed under a Creative Commons Attribution 4.0 International License

Chapter 16

SECURING BLUETOOTH LOW ENERGY LOCKS FROM UNAUTHORIZED ACCESS AND SURVEILLANCE

Anthony Rose, Jason Bindewald, Benjamin Ramsey, Mason Rice and Barry Mullins

Abstract This chapter describes several vulnerabilities that affect commercial and residential Bluetooth Low Energy security devices and outlines methods for exploiting plaintext, obfuscated and hard-coded passwords, brute forcing passwords and hashes, fuzzing commands and performing man-in-the-middle attacks. Evaluations reveal that 75% of the tested security and access control systems have vulnerabilities that grant unauthorized access. In addition to obtaining access, malicious actors can extract sensitive information that can be used to develop patterns of human behavior. This chapter discusses five solutions for preventing or mitigating Bluetooth Low Energy security breaches, most of which involve minimal implementation overhead on the part of developers.

Keywords: Bluetooth Low Energy, access control, locks, vulnerabilities, security

1. Introduction

Bluetooth Low Energy, also marketed as Bluetooth Smart, is a wireless protocol designed for interconnecting Internet of Things (IoT) devices. The Internet of Things is an expanding market that includes linkages from home automation systems to industrial control systems and is expected to grow to more than $19 trillion devices by 2020 [20]. Bluetooth Low Energy devices, with nearly 8.2 billion already in use worldwide, currently constitute more than one-third of all Internet of Things devices [3]. Meanwhile, Internet of Things devices are becoming increasingly intertwined with water, power, emergency services, health care, agriculture, transportation and security systems [15].

Physical security relies on access control to manage admittance to sensitive locations. Typical access control implementations involve the use of personal identification numbers (PINs), radio frequency identification (RFID), public

key infrastructures and biometrics [5]. These solutions limit the ability of an organization to control credentials (e.g., granting or revoking access on demand). In some cases, access revocation may be impossible or expensive without revoking the credentials of all the users. A major appeal of Bluetooth Low Energy and other wireless systems is that access can be centrally managed. This requires authentication between the user and organization database, which helps eliminate the need to manage devices independently.

Several vendors have released security systems that use Bluetooth Low Energy locks to grant access to server rooms, power plants, water treatment facilities, manufacturing plants and ATMs. Onity [18] offers automation, manufacturing and security products; more than one million of its Bluetooth locking systems are being used in 115 countries.

The growth of Bluetooth within the Internet of Things paradigm has motivated the evaluation of commercial lock systems. Tests conducted as part of this research have revealed vulnerabilities in thirteen of the seventeen evaluated Bluetooth Low Energy locks. This chapter describes the vulnerabilities and proposes implementation guidance for securing the devices.

2. Bluetooth Low Energy

Bluetooth is an umbrella term that covers two completely different protocols: (i) Bluetooth Classic (BTC); and (ii) Bluetooth Low Energy (BLE). Bluetooth Classic focuses on sending the maximum amount of data without regard to power consumption (e.g., music streaming and data storage). Conversely, power saving is a top priority for Bluetooth Low Energy devices. Bluetooth Low Energy offers an interface for low-data-rate devices (e.g., temperature monitors and door locks). Bluetooth Low Energy is also designed to provide a secure and robust wireless communications mechanism that requires minimal energy at data rates up to 1 Mbps [12].

The Bluetooth Low Energy connection process differs from Bluetooth Classic by limiting the device transmission time, thereby minimizing the expended energy. Devices advertise themselves on three channels that are dispersed across the 2.4 GHz band to avoid interference from IEEE 802.11 wireless local area networks (WLANs) [24]. A user connects to a device on an advertising channel to initiate a connection. Bluetooth Low Energy operates under a master/slave model, where the master is typically the user (e.g., phone or tablet) and the slave is the device that awaits a connection (e.g., lock, thermostat or heart rate monitor). Bluetooth Low Energy devices are split into two categories depending on their function: (i) client; and (ii) server. The client is the master in most cases, while the slave is the server. Common Bluetooth Low Energy operations include read, write, notify and indicate, which push or pull data between the client and server through the Generic Attribute Profile (GATT).

The Generic Attribute Profile is constructed as a hierarchy in which the profile is at the top level and is composed of a series of services. Services are collections of characteristics that represent the behavior of a device. For example, a service could be listed as a blood pressure monitor or heart rate monitor

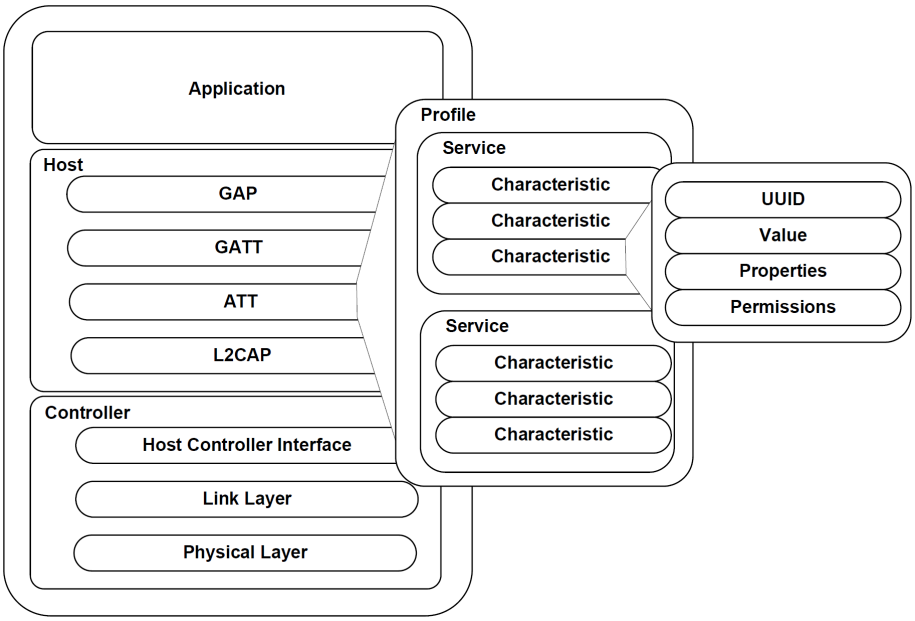


Figure 1. Bluetooth Low Energy stack hierarchy.

for medical devices or temperature readings for thermostats. Characteristics fall into a few different categories below a service. Each service has a universally unique identifier (UUID), value, properties (e.g., read, write, notify and indicate) and permissions. The UUID is a 16-bit or 128-bit identifier used by a manufacturer to specify custom services; however, some UUIDs are universally used across manufacturers. Finally, descriptors fall under characteristics and contain configuration flags and metadata that a manufacturer may desire to share. Figure 1 presents the Bluetooth Low Energy hierarchy.

3. User Behavioral Analytics

User behavior analytics (UBA) detects anomalies that indicate potential insider threats and targeted attacks by tracking and analyzing user behavior. Defensive mechanisms employ user behavior analytics to help prevent attacks. This research proposes an offensive approach that leverages user behavior analytics.

Reconnaissance is one of the most important stages in penetration testing because it helps acquire detailed knowledge of a target prior to an attack [16]. During this phase, a target is continuously monitored for all activity.

Time is critical when it comes to gaining information about a target and acting on it. Minimizing the time spent between target reconnaissance and infiltration greatly enhances the likelihood of an attack being successful [21].

Table 1. Exploited Bluetooth Low Energy devices by type.

Device Name	Type	A	B	C	D	E	F
Safetech Quicklock Doorlock	Deadbolt	✓		✓			✓
Vians Doorlock	Deadbolt		✓				✓
Lagute Sciener Doorlock	Deadbolt		✓				✓
Okidokeys	Deadbolt				✓		✓
Poly-Control Danalock	Deadbolt					✓	✓
Ceomate Doorlock	Deadbolt	✓	✓				✓
Safetech Quicklock Padlock	Padlock	✓		✓			✓
Elecycle EL797	Padlock		✓				✓
Elecycle EL797G	Padlock		✓				✓
Mesh Motion Bitlock	Padlock						✓
iBlulock	Padlock	✓		✓			✓
Safetech Gunbox 2.0	Gun Safe	✓		✓			✓
Plantraco Phantomlock	Cabinet Lock	✓		✓			✓

In the past, an attacker would physically monitor a target to gain information; however, attackers can now leverage information present in Bluetooth Low Energy devices in developing attacks.

A number of devices store system logs that contain valuable user behavior analytics information (e.g., user names and timestamps). Applying statistical analysis methods to system logs generates meaningful information that can be leveraged in attacks. For example, user behavior patterns can be inferred, which would provide the ideal times to inject malicious code and avoid detection.

4. Bluetooth Security Vulnerabilities

A wide variety of attacks against Bluetooth Low Energy devices have been developed; most of them exploit vulnerabilities inherent in the protocol or errors in vendor implementations. An analysis of seventeen Bluetooth Low Energy locks reveals that thirteen devices were vulnerable to eight exploits [23]. Table 1 lists the exploited Bluetooth Low Energy devices by type. Note that A denotes plaintext passwords, B password obfuscation, C brute forcing passwords and hashes, D command fuzzing, E hard-coded passwords and F man-in-the-middle attacks. This section discusses the vulnerabilities present in the tested Bluetooth Low Energy devices and how an adversary may exploit them.

The hardware required for Bluetooth Low Energy eavesdropping is affordable. Higher-end devices such as the HackRF One and Ubertooth One are more expensive alternatives that have high power amplifiers and detachable antennas. Replacing an antenna increases the operational range of a device. Increased sniffer range eliminates the need to be near a target to obtain its credentials. Pairing a long-range sniffer with a high power Bluetooth adapter



Figure 2. Reverse engineered Safetech command structure.

(e.g., Sena UD-100) enables commands to be transmitted at distances up to half a mile.

The National Institute of Standards and Technology (NIST) [19] has reported that flawed Bluetooth security implementations are highly susceptible to wireless attacks (e.g., denial-of-service (DoS), eavesdropping, man-in-the-middle attacks, message modification and resource misappropriation). These attacks on Bluetooth systems can be leveraged by adversaries to obtain unauthorized access to sensitive information.

4.1 Plaintext Passwords

A plaintext password is stored or transmitted in a readable format and offers no protection to the user or device. This vulnerability is very common [1, 22] and it enables an adversary to eavesdrop on a conversation through specialized hardware or to embed software that monitors the host controller interface (HCI) traffic. Stolen plaintext passwords can be used to gain access to secure facilities, change administrative privileges or obtain system logs.

Figure 2 illustrates the ease with which a plaintext password can be used in an attack. The Safetech command structure has not been published and was discovered by reverse engineering. The command structure is used in several Safetech Bluetooth Low Energy devices (e.g., door locks, padlocks and safes). The first byte is an opcode that specifies if a device should read the password (`00`) or change the user password (`01`). A user password can be modified merely by changing the first byte and placing the current user password into the next four bytes. The final four bytes are then used to set the new user password. Note that the password for this type of device is limited to numbers.

4.2 Password Obfuscation

Obfuscated passwords provide more protection than plaintext passwords, but they still constitute a major security risk. An obfuscated password uses hashing or encryption to reduce the risk of exposure [4].

The problem with using an obfuscated password is that it can be recorded and replayed to a Bluetooth Low Energy lock. Replaying a password enables an adversary to gain access to the lock without knowing the password. Moreover, if the device uses the same hashing algorithm for the password every time, an adversary can gain access at any time using the sniffed obfuscated password. However, depending on the security implementation, some high-level functions may not be accessible. This could deter an attack that requires a password to gain access to high-level functions.

Table 2. Time expected to brute force passwords.

Available Characters	Password Length	Password Possibilities (millions)	Expected Completion Time (years)
10	8	10^2	0.03
10	16	10^{10}	3.17
128	8	7.2×10^{10}	22.83
128	16	5.1×10^{27}	1.61×10^{18}
256	8	1.8×10^{13}	5,475
256	16	3.4×10^{32}	1.06×10^{23}

Password obfuscation is implemented in the Ceomate door lock. Reverse engineering techniques applied to this product were unable to determine the proprietary hashing process. However, an attacker can still gain access to the device – without knowing the original password and hashing algorithm – merely by replaying the recorded hashed password.

4.3 Brute Forcing

Brute forcing involves submitting repeated guesses of a password or hash with the goal of gaining access [7]. This attack requires a guess of the plaintext password or obfuscated password. Obviously, if the number of password or hash possibilities are massive, a brute force attack is impractical. For example, a device that uses a six-digit PIN could be brute forced in hours, while an eight-character password would require nearly a month.

Table 2 shows the expected amount of time for brute forcing passwords based on the number of available characters and password length. Password length is more important than the number of available characters. Doubling the password length exponentially increases the number of password combinations, while doubling the available characters has a much smaller effect on the number of combinations. Timeouts between successive password attempts significantly reduce the speed of the overall attack.

It is important to use long PINs to protect devices. Assuming a speed of 6,000 attempts/min, a PIN that uses only numbers and has a maximum length of eight could be brute forced within twelve days. Factors that limit an attack include the rate at which the transmitting device sends packets and the speed at which the receiver translates the data. However, the speed of a brute force attack can be greatly increased if a web server is used to store the credentials.

4.4 Command Fuzzing

Command fuzzing occurs when an application accepts an invalid command that has been modified to mimic a valid command, potentially causing the



Figure 3. Reverse engineered Okidokeys command structure.

device to enter a new state [17]. This technique involves changing the individual bytes of a packet until the targeted application accepts the invalid command. The goal of fuzzing is to force a device to move into an unstable state in which it behaves in a manner different from what was designed. For example, a lock may go into an error state as a result of a fuzzed command and open without proper authentication. Opening in an error state is usually a design decision made for reasons of fire safety. A device may default to a locked state when entering an error state, but these designs are typically implemented in prison lock systems where locking by default is required. Fuzzing can be problematic when designers implement proprietary encryption. Well-established encryption methods (e.g., Advanced Encryption Standard (AES)) have been proven to offer secure communications channels [14].

Figure 3 shows the reverse engineered Okidokeys command structure (in bytes) at the host controller interface level and the fuzzed packet. Okidokeys describes the implementation as using a highly-secure patented cryptographic solution that offers the same protection as 256-bit AES. However, experiments have revealed that the generated keys are not unique. Specifically, the keys have patterns that are not encountered in AES and other encryption algorithms. This prompted the fuzzing of a previously-valid command that forced the lock to move to an error state in which it opened.

4.5 Hard-Coded Passwords

Hard-coded passwords, where designers leave passwords in applications, are the result of poor programming practices. Such passwords are encountered in more than 40% of Android applications [8]. However, hard-coded passwords are difficult to find because they require applications to be decompiled into readable code. Another method for capturing an administrative password is to implant malware with a keystroke logger on a target device. Hard-coded passwords offer an attacker the ability to gain access to developer options inside of an application and to bypass the built-in security controls.

The easiest method for finding the hard-coded passwords to a Bluetooth Low Energy device is to decompile an Android application package (APK). In general, an attacker would decompile an application after the Android application package has been removed from the device. Programs such as Bytecode Viewer offer a user-friendly environment to reverse engineer Android application pack-

```
public String getPassword(){
    Cursor localCursor = getReadableDatabase().query("USER_TABLE"),
        DatabaseContract.UserTableColums, null, null, null, null, null, null);
    if (localCursor == null) {
        return "";
    }
    if (localCursor.moveToFirst())
    {
        byte[] arrayOfByte = xor(new String(Base64.decode(
            localCursor.getString(local.getColumnIndex("password"))
            .getBytes(),1).getBytes(),"thisisthesecret".getBytes());
        localCursor.close();
        return new String(arrayOfByte);
    }
    return "";
}
```

thisisthesecret

Figure 4. Hard-coded password found in a Danalock.

ages into readable Java code. This readable code is parsed for keywords to reveal hard-coded passwords, developer comments and other valuable information.

Figure 4 shows a hard-coded password found by decompiling the Danalock application. The plaintext password is stored in a table with the passphrase `thisisthesecret`. Having discovered the password, the adversary can gain access to the lock. Decompilation may also reveal the method of encryption and other information that is hidden. This provides additional opportunities to compromise the system.

4.6 Man-in-the-Middle Attack

A man-in-the-middle attack occurs when two devices are unknowingly connected to a third device that relays information between the two communicating devices [2]. This attack is effective when devices use unauthenticated connections, enabling an attacker to intercept as well as modify and inject fake information or commands. Several tools have been developed for implementing man-in-the-middle attacks (e.g., GATTacker and BTLEjuice). Two attacks that leverage the man-in-the-middle concept are: (i) rogue device attack; and (ii) relay attack.

Rogue Device Attack. In a rogue device attack, an attacker impersonates a target device with the intention of convincing the other communicating device that the rogue device is, in fact, the target device. The majority of applications do not properly authenticate with devices before sending commands, enabling an attacker to clone the target device and send advertisements. The user application initiates a connection after it receives the cloned device advertisement. The user application then sends commands to the cloned device assuming it to be the target device. These commands include passwords and nonces that could be used by an attacker to gain access to the target device.

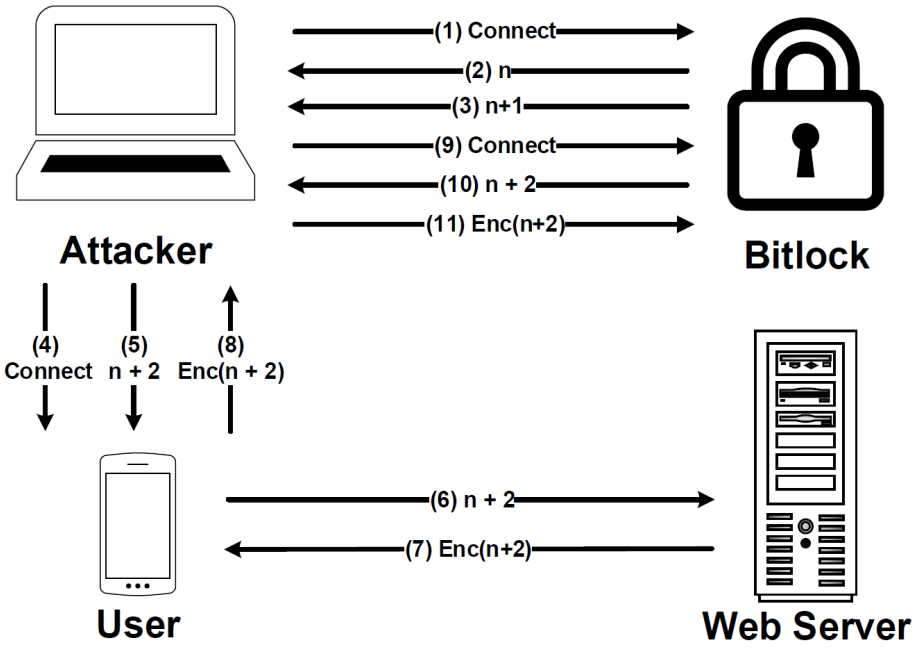


Figure 5. Sequence diagram of a rogue device attack on a Mesh Motion Bitlock.

A nonce is a random number that is used only once and protects a communications connection from a replay attack.

A rogue device attack can be used to exploit a web server that stores user credentials. A user application would be unable to distinguish between the true device and the cloned device, enabling an attacker to steal credentials from the web server. All that the attack requires is for the cloned device to interact with the user application.

Figure 5 shows a rogue device attack on a Mesh Motion Bitlock. This product does not use a plaintext password; however, it has a predictable nonce that enables an adversary to collect credentials and use them to control the lock. The user in this case must have an Internet connection in order to receive credentials from the web server. An attacker connects to the lock and sends invalid credentials with the intention of receiving the current nonce value. The value is sent with the initial connection and is incremented by one when receiving invalid credentials. The user is unaware that his/her device is connected to a spoofed lock when the next nonce is received. During the connection, the user forwards the nonce to the web server and receives the credentials in return. Finally, the credentials are sent from the user to the spoofed lock.

Because the web server trusts the user application, the attacker is not limited to receiving just one set of credentials. In fact, the attacker can flood the user application with nonces with the goal of creating a table to gain permanent

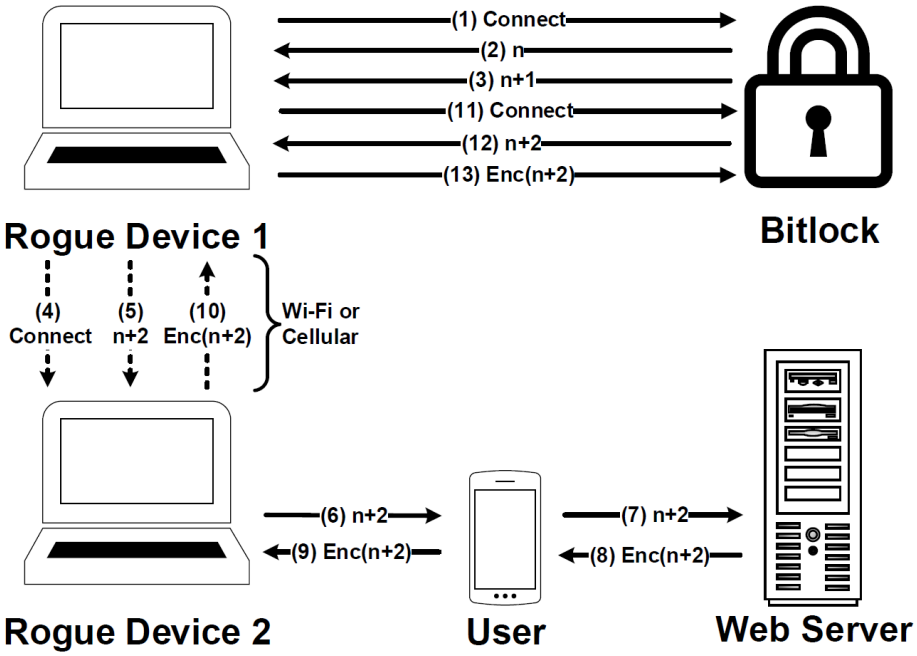


Figure 6. Sequence diagram of a relay attack on a Mesh Motion Bitlock.

access to the device. The attacker can use the table of credentials to open the lock at any time.

Relay Attack. A relay attack is similar to a rogue device attack, but it is designed specifically for scenarios where the nonces are truly random and a rogue device attack is not possible. In this attack, an attacker impersonates a target device and forces the user to communicate via a bridge to the attacker's device. This enables the attacker's device to impersonate the target device and trick the user into communicating with the attacker.

Figure 6 shows how two rogue devices can create a relay attack. Rogue Device 1 connects to the user while Rogue Device 2 connects to the target device. The target device generates a nonce and sends it to the cloned user (Rogue Device 1). Rogue Device 2 connects to the user and impersonates the target device.

After the two rogue devices are in place, a bridge is established using Wi-Fi, cellular or some other means. The bridge supports communications between the rogue devices and facilitates the hand-off of the nonce during the attack. Rogue Device 2 sends the nonce to the user, who then forwards the nonce to the web server to generate the credentials. This phase of the attack mirrors the rogue device attack discussed above. Finally, the credentials that the user unexpectedly generated are passed from Rogue Device 2 back to Rogue Device 1. These credentials are used by the attacker to gain access to the target device.

The danger of a relay attack is that a user can be anywhere as long as a rogue device is nearby to impersonate the target device. This type of attack can be used to target a variety of devices because large organizations require many access points and rely on a central server to handle user credentials.

5. Attack Scenario

Numerous devices store system logs of user activity with information such as user names, permissions and timestamps. An attacker can extract the system logs from locks and analyze the information to construct a profile of activity in a facility. The attacker can use the behavior patterns to gain insight into the organization's inner workings.

This section presents an attack scenario involving a manufacturing facility that uses several Bluetooth Low Energy locks for access control. The Bluetooth Low Energy system has a central server that manages credentials, requiring employees to authenticate via an application installed on their mobile devices. The simulated data in the scenario mimics real data found in employee devices. A proven method for extracting real data is presented, but simulated data is still required to meet the goals of the scenario.

The scenario involves the following steps:

- The attacker connects to a security door lock and scans for all services, characteristics and descriptors.
- The attacker uses the scanned information to construct an identical Bluetooth Low Energy device. The cloned device is used to impersonate the lock and convince the user application to transmit its credentials.
- Concurrently, the attacker uses a second device near the lock to impersonate the user. This setup mirrors the relay attack discussed above.
- The attacker relays information (e.g., nonces and credentials) from the user to the lock via the relay attack. The relay attack provides access to the Bluetooth Low Energy lock for exploitation.
- The attacker accesses developer and administrator privileges to create additional accounts and download system logs.
- The attacker applies user behavior analytics on the system logs to reveal behavior patterns (Figures 7–9). Analysis of the logs provides detailed information about facility operations by highlighting user activity based on the time and/or day of the week.
- The attacker determines the best time to infiltrate the facility based on the analysis.

The analyzed data can provide important information when cross referenced against employee public records. Specifically, user activity may be analyzed in three formats: (i) all user activity by day of week and time of day; (ii)

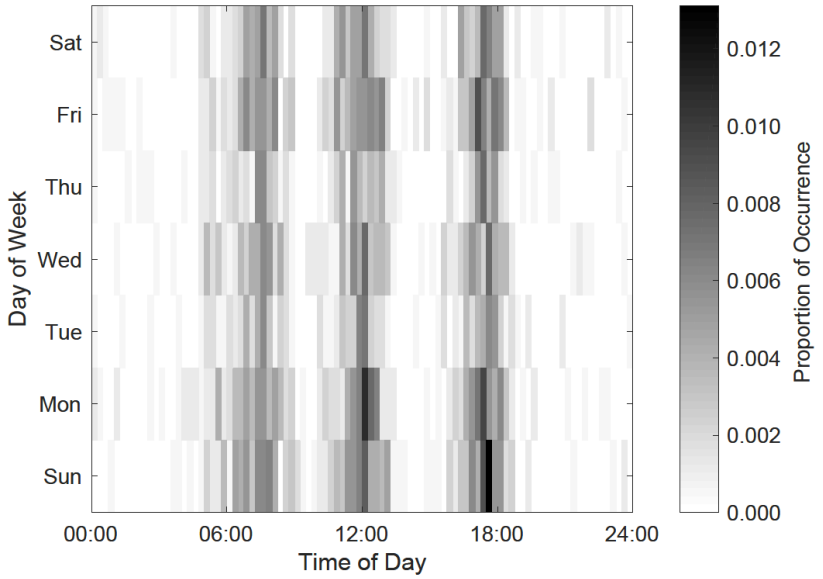


Figure 7. Heat map of historic weekday activity compared with time of day activity.

individual user activity by time of day; and (iii) individual user activity by day of the week.

The heat map in Figure 7 demonstrates that the overall user activity can be determined by comparing user activity to the time of day that users are active. User activities corresponding to entering and leaving the facility are indicated in the heat map, where the darker shades represent higher levels of user activity.

The heat map in Figure 8 highlights the user activity during historical days worked. Finally, the heat map in Figure 9 breaks down user activity on any given day by the time of day. Analyzing this information enables an attacker to determine the ideal day and time to access the facility. Additional information can also be inferred, such as odd activity at specific times of the day for specific users; this may indicate specific tasks (e.g., maintenance).

6. Mitigation Techniques

Many mitigation techniques have been proposed for combating Bluetooth attacks. Table 3 lists several mitigation techniques and the vulnerabilities against which they protect. Note that A denotes plaintext passwords, B password obfuscation, C brute forcing passwords and hashes, D command fuzzing, E hard-coded passwords and F man-in-the-middle attacks. The last two columns of Table 3 rank the implementation and maintenance difficulty of each proposed solution.

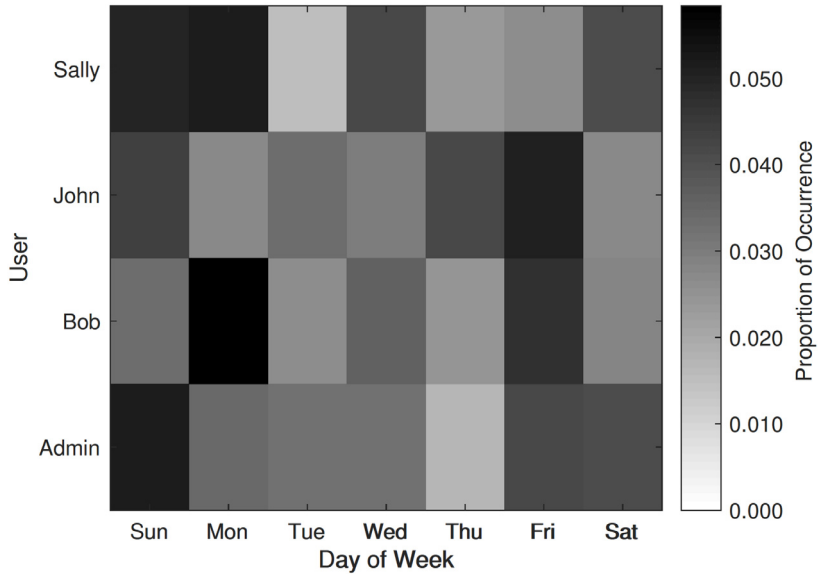


Figure 8. Heat map of historic user activity compared with weekday activity.

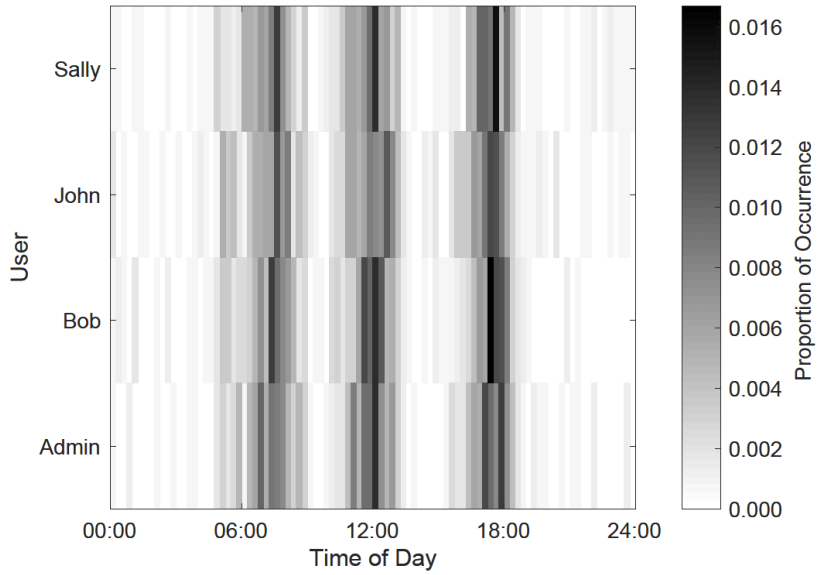


Figure 9. Heat map of historic user activity compared with time of day activity.

6.1 Pairing and Bonding

Pairing and bonding protect against malicious eavesdroppers. Two processes occur during the initial connection. The first step is pairing, which involves an

Table 3. Mitigation techniques and their difficulty.

Mitigation Techniques	A	B	C	D	E	F	Difficulty of Use	Difficulty of Maintenance
Pairing and Bonding	✓	✓	✓	✓			Low	Low
App Layer Encryption	✓	✓	✓	✓			Medium	Medium
Two-Way Authentication	✓	✓	✓	✓			Medium	Low
Geofencing						✓	Medium	High
BLE Guardian						✓	High	High

exchange of security features and capabilities. This step begins with the client and establishes the types of input and output mechanisms that exist in the device and dictates the type of bonding. Bonding occurs after pairing and the keys have been generated and exchanged. Bonding is a more permanent encryption method that saves the key for use in future connections [25]. When devices are bonded, they can encrypt their connections without having to exchange keys. Bluetooth Low Energy uses AES-CCM encryption after the key exchange process has been completed.

Bluetooth Low Energy uses a secure simple pairing model where devices use one of the following pairing modes:

- **Just Works:** This mode offers little protection. The mode sets the temporary key to all zeroes, enabling any eavesdropper to immediately guess the temporary key. The Bluetooth Special Interest Group documentation [2] notes that Just Works provides no protection against eavesdropping and man-in-the-middle attacks.
- **Passkey Entry:** This mode requires the user and device to use the same six-digit PIN as the temporary key, while the rest of the 128-bit AES key is padded with zeroes. Passkey entry provides only slightly more protection against eavesdropping and man-in-the-middle attacks than Just Works. In fact, previous research has shown that it can be brute forced [24].
- **Numeric Comparison:** This mode is similar to passkey entry, except that both devices input a six-digit PIN independently. This greatly reduces the probability of brute forcing both the PINs.
- **Out-of-Band Communications:** This mode employs the full 128-bit temporary key that is communicated over a non Bluetooth Low Energy channel, typically using near field communications (NFC) technology. Another method is to send the temporary key over Bluetooth Classic because most devices are already equipped to handle both Bluetooth Low Energy and Bluetooth Classic. However, this method is not typical and was only implemented in one of the seventeen devices tested in this research. The use of an out-of-band channel is extremely important and is the best option when using secure simple pairing [10].

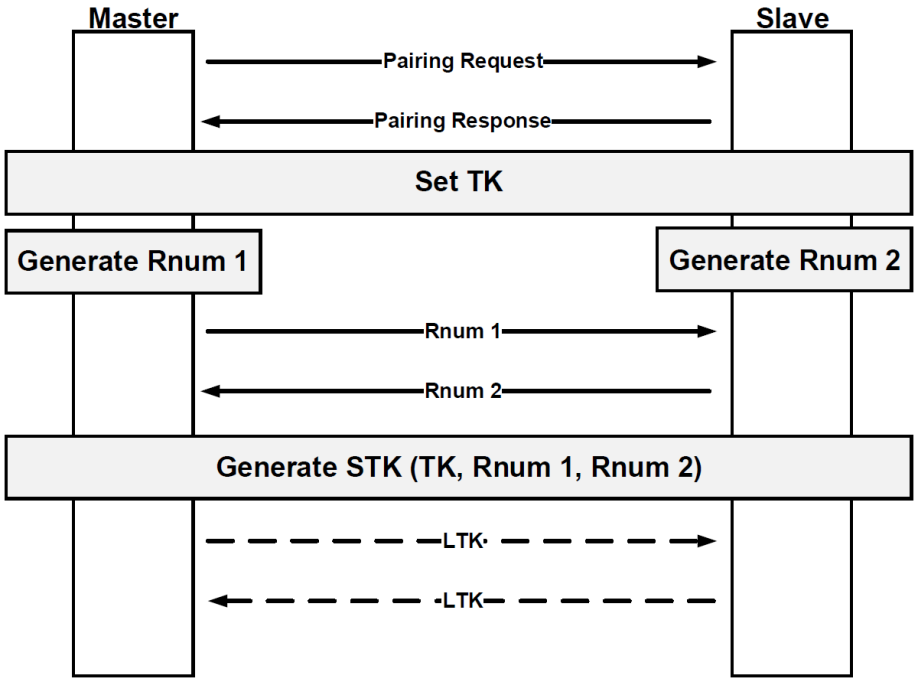


Figure 10. Bluetooth Low Energy Version 4.1 long-term key generation.

The numeric comparison technique is a simple solution for manufacturer implementation because the Bluetooth protocol already supports this type of authentication. However, the technique is only practical if developers use the key exchange improvements specified in Bluetooth Version 4.2. Unfortunately, all the developers whose products were investigated have not used these improvements. Therefore, a new key generation process is incorporated in Bluetooth Version 4.2. This process uses Elliptic Curve Diffie-Hellman (ECDH) key generation and implements new procedures for key generation.

Bluetooth Version 4.1 Link Layer Encryption. Pairing and bonding protect against compromises of plaintext and obfuscated passwords as well as brute forcing and fuzzing attacks. An added feature when using pairing and bonding is the ability to establish link layer encryption. The encryption method used in versions 4.0 and 4.1 is derived from the devices being paired initially and uses the long-term key as shown in Figure 10.

The process for generating the long-term key begins with the temporary key determined through the pairing modes mentioned above (i.e., Just Works, passkey entry, numeric comparison and out-of-band communications). The temporary key is used to encrypt the short-term key, which is generated using the temporary key and two random numbers from the master and slave. Finally,

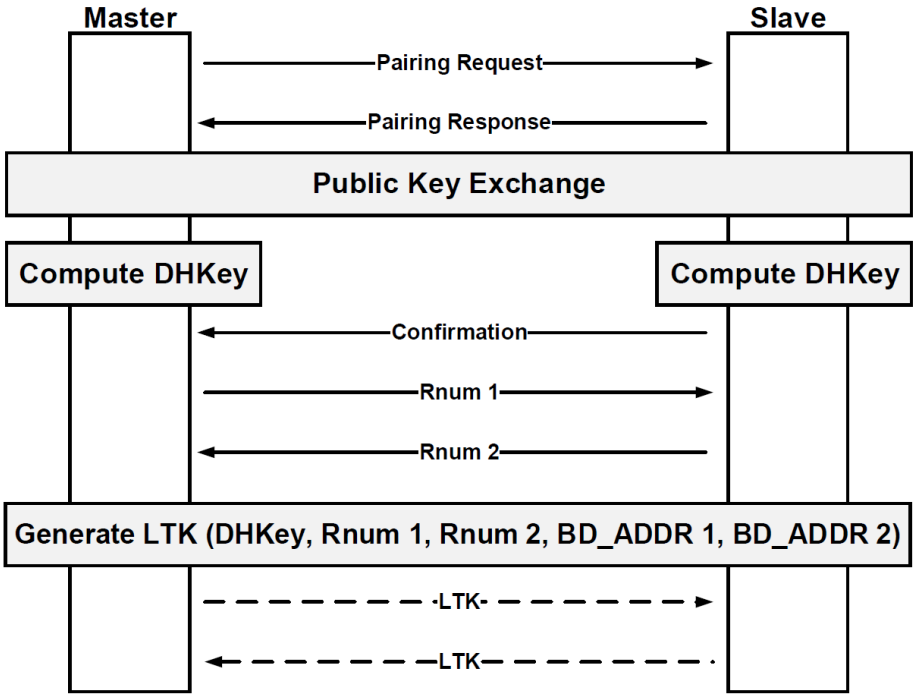


Figure 11. Bluetooth Low Energy Version 4.2 long-term key generation.

the short-term key is used to encrypt the long-term key, which is saved and used for all other communications.

Bluetooth Version 4.2 Link Layer Encryption. Bluetooth Low Energy Version 4.2 no longer uses a short-term key; instead, it uses a key derived from an ECDH key. Figure 11 shows the new key generation method. First, a pairing request occurs, which establishes the key generation method. A public key is exchanged to initiate the long-term key generation process. After the public key is exchanged, each device independently computes an ECDH key using the public key of the other device. Next, the slave computes a confirmation message that the master uses to check against its own key. If the check succeeds, the master sends a random number to the slave. The slave responds with a random number, which initiates the long-term key generation process.

The long-term key requires five parameters: (i) ECDH key; (ii) random number 1 from the master; (iii) random number 2 from the slave; (iv) Bluetooth device address of the master; and (v) Bluetooth device address of the slave. The major change in the protocol is that the ECDH key is never transmitted and is computed independently to protect against eavesdropping. Passive eavesdropping is no longer possible in version 4.2 because of the difficulty of guessing the private key [9]. The other information needed to generate the long-term

key in version 4.2 comprises the random numbers sent by each device and their Bluetooth device addresses (BD_ADDR). The connection is encrypted and authenticated after the long-term key is established and this key is used for all future connections.

6.2 Application Layer Encryption

Application layer encryption is one of the most popular methods for securing Bluetooth Low Energy devices. The rationale for application layer encryption is that it does not require new devices to be paired; instead, it relies on the user and device to establish keys to encrypt and decrypt credentials. Application layer encryption can be more complicated than using the standard pairing offered by Bluetooth Low Energy due to the difficulty of managing keys [11]. However, the additional complexity of application layer encryption adds an extra layer of security when it is combined with link layer encryption. Good cryptographic practices (e.g., true random number generation and non-proprietary encryption algorithms) are encouraged for vendor implementations. Application layer encryption protects against attacks on plaintext passwords and obfuscated passwords, as well as brute forcing and fuzzing.

6.3 Two-Way Authentication

Two-way authentication protects against a rogue device attack by forcing the user and device to not (immediately) trust the connection. This method does not use link layer encryption. Instead, public/private keys are used for authentication by devices. For example, the public key of the user is used by the lock to encrypt the nonce $N1$, which is sent to the user. At the same time, the lock sends a plaintext nonce $N2$ to the user. The user decrypts $N1$ with his/her private key and encrypts $N2$ with the public key of the lock. Next, the user replies to the lock with the decrypted $N1$ and encrypted $N2$. Using an asynchronous encryption method ensures that the user and lock have public and private keys, and prevents a rogue device from impersonating a legitimate device. A rogue device would not be able to attack a connection without the private key of one of the devices and the public key of the other device. Thus, two-way authentication protects against attacks on plaintext passwords and obfuscated passwords, as well as brute forcing, fuzzing and man-in-the-middle attacks.

6.4 Geofencing

Geofencing protects against unauthorized access by requiring a user to be within a specific distance of designated GPS coordinates in order to request credentials from a web server. Essentially, a virtual fence is created around a device, where a user must be within a set distance (usually a few feet) to gain access. Geofencing prevents cloned devices from tricking users into providing

their credentials. Thus, geofencing protects against rogue device and relay attacks.

The August Lock, a Bluetooth Low Energy lock, implements geofencing. However, geofencing is best combined with other mitigation techniques because it offers no protection against eavesdropping when a user is within the geofence perimeter. Other attacks are also possible in the case of the August Lock. These include replacing the firmware with malicious code or gaining access to developer-only features that should have been removed before the application was released [13].

6.5 Bluetooth Low Energy Guardian

Bluetooth Low Energy Guardian protects user privacy using an administrative program to control which entities can discover, scan and connect to a device [6]. Bluetooth Low Energy Guardian also defends against advertisements. Most man-in-the-middle attacks leverage advertisement packets that do not provide privacy and security protection. However, Bluetooth Low Energy Guardian controls advertisement packets via reactive jamming and by managing the connection request approval process. These protections are required because true advertisement packets are shielded from passive eavesdropping.

The implementation of Bluetooth Low Energy Guardian requires an Ubertooth One in addition to the device being protected. An advantage of this approach is that it actively prevents attacks on a device compared with a standard encryption approach. No additional implementation is required on the part of the device manufacturer and the approach can be used in conjunction with other techniques to enhance security. The downside of the approach is that it requires additional hardware, which may not be practical. Bluetooth Low Energy Guardian protects against man-in-the-middle, rogue device and relay attacks.

7. Conclusions

This research has sought to enhance the security of Bluetooth Low Energy devices. Thirteen of the seventeen devices subjected to testing have vulnerabilities that can be mitigated using the security solutions presented in this chapter. Countermeasures to Bluetooth Low Energy attacks require minimal development and implementation efforts on the part of device manufacturers. The existing pairing and bonding feature of Bluetooth Low Energy provides adequate security to defeat basic attacks. Sophisticated mitigation techniques are expensive to implement, but they offer additional protection against advanced attacks. An alternative protection method is to shield device activity as in the case of Bluetooth Low Energy Guardian, but this requires significant implementation and maintenance efforts.

Note that the views expressed in this chapter are those of the authors and do not reflect the official policy or position of the U.S. Air Force, U.S. Army, U.S. Department of Defense or U.S. Government.

References

[1] R. Baldwin, Researcher finds huge security flaws in Bluetooth locks, *Engadget*, August 10, 2016.

[2] Bluetooth Special Interest Group, Specification of the Bluetooth System, Covered Core Package Version 4.2, Master Table of Contents and Compliance, Specification Volume 0, Kirkland, Washington, 2014.

[3] Bluetooth Special Interest Group, Bluetooth 5 quadruples range, doubles speed, increases data broadcasting capacity by 800%, Kirkland, Washington, June 16, 2016.

[4] S. Boonkrong and C. Somboonpattanakit, Dynamic salt generation and placement for secure password storing, *International Journal of Computer Science*, vol. 43(1), 2016.

[5] Entrust, Advanced Solutions for Critical Infrastructure Protection: Complying with the North American Electric Reliability Corporation Critical Infrastructure Protection Standards, Dallas, Texas, 2012.

[6] K. Fawaz, K. Kim and K. Shin, Protection privacy of BLE device users, *Proceedings of the Twenty-Fifth USENIX Security Symposium*, pp. 1205–1221, 2016.

[7] N. Flor and H. Shannon, Technology corner: Brute force password generation – Basic iterative and recursive algorithms, *Journal of Digital Forensics, Security and Law*, vol. 6(3), pp. 79–85, 2011.

[8] S. Gold, Android, a secure future at last? *Engineering and Technology*, vol. 7(2), pp. 50–54, 2012.

[9] K. Haataja, K. Hypponen, S. Pasanen and P. Toivanen, *Bluetooth Security Attacks: Comparative Analysis, Attacks and Countermeasures*, Springer-Verlag, Berlin Heidelberg, Germany, 2013.

[10] K. Haataja and P. Toivanen, Two practical man-in-the-middle attacks on Bluetooth secure simple pairing and countermeasures, *IEEE Transactions on Wireless Communications*, vol. 9(1), pp. 384–392, 2010.

[11] D. Herrmann, *Complete Guide to Security and Privacy Metrics: Measuring Regulatory Compliance, Operational Resilience and ROI*, Auerbach Publications, Boca Raton, Florida, 2007.

[12] R. Heydon, *Bluetooth Low Energy: The Developer's Handbook*, Pearson Education, Upper Saddle River, New Jersey, 2013.

[13] Jmaxxz, Backdooring the front door: Hacking a "perfectly secure" smart lock, presented at *DEF CON 24*, 2016.

[14] C. Kaufman, R. Perlman and M. Speciner, *Network Security: Private Communication in a Public World*, Prentice Hall, Upper Saddle River, New Jersey, 2002.

[15] National Security Telecommunication Advisory Committee, NSTAC Report to the President on Secure Government Communications, Washington, DC, 2013.

[16] T. O'Connor, *Violent Python: A Cookbook for Hackers, Forensic Analysts, Penetration Testers and Security Engineers*, Syngress, Waltham, Massachusetts, 2013.

[17] P. Oehlert, Violating assumptions with fuzzing, *IEEE Security and Privacy*, vol. 3(2), pp. 58–62, 2005.

[18] Onity, Electronic Locks, Salem, Oregon (`en.onity.com/products/Pages/Electronic-Locks.aspx`), 2016.

[19] J. Padgette, K. Scarfone and L. Chen, Guide to Bluetooth Security, NIST Special Publication 800-121, Revision 1, National Institute of Standards and Technology, Gaithersburg, Maryland, 2012.

[20] G. Press, Internet of Things by the numbers: Markets estimates and forecasts, *Forbes*, August 22, 2014.

[21] B. Ramsey, B. Mullins, W. Lowder and R. Speers, Sharpening the stinger: Tuning KillerBee for critical infrastructure warwalking, *Proceedings of the IEEE Military Communications Conference*, pp. 104–109, 2014.

[22] B. Ramsey, B. Mullins, R. Speers and K. Batterton, Watching for weaknesses in wild WPANs, *Proceedings of the IEEE Military Communications Conference*, pp. 1404–1409, 2013.

[23] A. Rose and B. Ramsey, Picking Bluetooth Low Energy locks from a quarter mile away, presented at *DEF CON 24*, 2016.

[24] M. Ryan, Bluetooth: With low energy comes low security, *Proceedings of the Seventh USENIX Conference on Offensive Technologies*, 2013.

[25] K. Townsend, C. Cufi, Akiba and R. Davidson, *Getting Started with Bluetooth Low Energy: Tools and Techniques for Low-Power Networking*, O'Reilly Media, Sebastopol, California, 2014.