Tamarin-based Analysis of Bluetooth Uncovers Two Practical Pairing Confusion Attacks*

Tristan Claverie[1,2,3], Gildas Avoine[2,3], Stéphanie Delaune[3] and José Lopes Esteves[1]

[1] Agence Nationale de la Sécurité des Systèmes d'Information (ANSSI), Paris, France
[2] INSA de Rennes, France
[3] Univ Rennes, CNRS, IRISA, France

Abstract. This paper provides a Tamarin-based formal analysis of all key-agreement protocols available in Bluetooth technologies, *i.e.*, Bluetooth BR/EDR, Bluetooth Low Energy, and Bluetooth Mesh. The automated analysis found several unreported attacks, including two attacks that exploit the confusion of Pairing modes, which occurs when a communicating party uses the Secure Pairing mode while the other one uses the Legacy Pairing mode. They have been validated in practice using off-the-shelf implementations for the genuine communicating parties, and a custom BR/EDR machine-in-the-middle framework for the attacker. Our attacks have been reported by Bluetooth SIG as CVEs.

1 Introduction

Bluetooth technologies are increasingly used worldwide as ways to transmit data over-the-air. In 2021, 4.7 billion Bluetooth devices were shipped according to the Bluetooth Special Interest Group (SIG) [24]. There are actually three distinct Bluetooth technologies: Bluetooth Basic Rate/Enhanced Data Rate (BR/EDR), Bluetooth Low Energy (BLE), and Bluetooth Mesh (BM). While the details differ, all of them aim at providing confidentiality, integrity, and authentication.

Many flaws have been discovered over the years in Bluetooth standards. Some of them are related to the use of improper cryptographic primitives [30–32], others are purely protocol-level flaws [1, 2, 15, 37, 40], and a few ones rely on incorrect implementations of cryptographic primitives [7, 18, 35]. The behaviour of Bluetooth stacks was also studied, especially on mobile platforms [3, 41, 42], revealing some vulnerabilities in implementations.

Bluetooth communication security mostly relies on the key agreement step, which can be performed using many different protocols and sub-protocols. This makes the security analysis highly complex. The *pairing confusion* introduced in [37] is an attack that exploits the interaction of two key-agreement protocols in Bluetooth. It consists in a scenario where an entity uses Protocol A while the other entity uses Protocol B, such that they are not aware of this protocol

* This work received funding from the France 2030 program managed by the French National Research Agency under grant agreement No. ANR-22-PECY-0006.

mismatch. Usually, such a mismatched interaction ends with a failure. However, for some protocol pairs, the attacker can exploit messages sent in Protocol A to break the security properties of Protocol B, and conversely.

Formal protocol verification is the process of abstracting a protocol to prove that the considered security properties hold. Tamarin [28] and ProVerif [8] are state-of-the-art tools that automatically perform this formal protocol verification. They have been used for verifying complex protocols such as TLS 1.3 [6,17] and 5G-AKA [16]. When their analyses complete, they grant either a formal proof that the considered security property hold, or an attack.

Several studies of protocol confusion have been performed for Bluetooth key agreements [23, 33, 40] using automated formal tools, but not in a systematic way. Although [23] and [33] consider some imperfect primitives, those representations are not accurate with regards to the current knowledge about Bluetooth protocols. As a result, most known attacks are not identified by those analyses.

Contributions. In this paper, comprehensive Tamarin models of all Bluetooth key-agreement protocols defined in BR/EDR, BLE, and BM are detailed. Those models are enhanced with representations of cryptographic imperfections that affect Bluetooth. In particular, they are used to systematically analyse pairing confusions in Bluetooth key agreements. Tamarin automatically identifies previously published attacks and identifies five new attacks, including four novel cases of protocol confusion. We highlight that the Bluetooth SIG assigned two CVEs for two of those attacks that defeat currently known mitigations against pairing confusions. To explore the practicality of these attacks, a BLE and a BR/EDR Machine-in-the-Middle (MitM) are implemented on the respective pairing methods of those technologies. To the best of our knowledge, this is the first practical MitM implementation on the BR/EDR pairing. Two additional attacks defeat proposed patches of BM Provisioning from the literature. A detailed research report of the work, including our Tamarin models, can be found in [14].

Outline. Section 2 provides an introduction to Bluetooth key-agreement protocols and their flaws. Section 3 details the Tamarin formal models developed for this study. The results, including new attacks and their implementations are described in Section 4 before being compared to the literature.

2 Background

In this section, we introduce two distinct Bluetooth technologies: Bluetooth Basic Rate / Enhanced Data Rate (BR/EDR) and Bluetooth Low Energy (BLE), respectively standardised in 1999 and 2010 [9]. Bluetooth Mesh (BM) is not described in this section, but a description can be found in [14]. BR/EDR is routinely used in audio devices (*e.g.*, earbuds, speakers) while BLE is commonly used in other smart devices (*e.g.*, watches). They have a similar security architecture. Both technologies try to grant confidentiality, integrity and authenticity of communications. Those properties rely on symmetric keys that are exchanged during a key agreement.

2.1 Key Agreement

In BR/EDR and BLE, the key agreement step is called *Pairing* and is performed between devices respectively called *Initiator* and *Responder*. To uniquely identify each protocol, two concepts are introduced. The term *Pairing mode* refers to the type of Pairing, it can be Legacy or Secure. The term *Pairing method* refers to the protocol name as standardized in the specification. The differences between the methods lie in the messages required to complete them and input/output capabilities of devices. Table 1 lists the Pairing protocols standardised. In this paper, a protocol is identified by a mode and a method, *e.g.*, Legacy JustWorks, Secure Out-of-Band (OOB), etc.

Table 1. BR/EDR and BLE Pairing protocols

	BR/EDR		BLE	
Pairing Mode	Legacy	Secure	Legacy	Secure
Pairing Method	PIN Pairing	JustWorks Passkey Entry Numeric Comparison Out-of-Band	JustWorks Passkey Entry Out of Band	JustWorks Passkey Entry Numeric Comparison Out-of-Band

For illustration purposes, we detailed below two protocols, the Legacy PIN Pairing protocol in BR/EDR, and the Legacy Passkey Entry protocol in BLE.

Legacy PIN Pairing for BR/EDR (Figure 1). Functions E_1, E_{21}, and E_{22} are defined in the specification [9](Vol 2, Part H, §6). The key agreement starts when the Initiator sends a nonce in_rand to the Responder ①. The user has to exchange a numeric code between devices, called the PIN ②. This PIN is used alongside in_rand and the Initiator address to derive K_{init}. K_{init} is used to mask two nonces $comb\_key_i$ and $comb\_key_r$ ③ which are used to derive the *Link Key* (LK) ④. According to the specification, the Pairing process is over once LK is created, but a mutual authentication procedure has to follow ⑤.

Legacy Passkey Entry for BLE (Figure 2). Functions c_1 and s_1 are defined in the specification [9](Vol 3, Part H, §2.2). The protocol starts with a Feature Exchange step ①, that is used to provide information about input-output capabilities, key size to use, etc. Then, the user has to exchange a numeric code between the devices ②. Typically, one device displays a code that the user enters in the other one. This code is used as a symmetric key in a commitment scheme ③. This step is used to authenticate the capabilities and respective addresses of the devices. Finally, nonces exchanged in step ③ are used to derive a *Short-Term Key* (STK) that is then used to encrypt the communication.

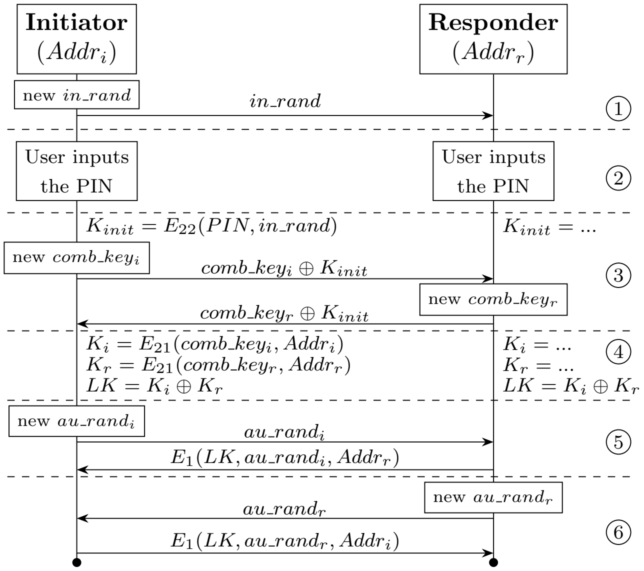

Fig. 1. BR/EDR Legacy PIN Pairing and Mutual Legacy Authentication

2.2 Tamarin Prover

The Tamarin prover [28] is a security protocol verification tool that supports both falsification and verification in the *symbolic model.* As usual in symbolic models, Tamarin represents the messages exchanged and computations as algebraic terms. Tamarin has already been successfully used to analyse many protocols, *e.g.*, TLS [17], WPA2 [19], and EMV [5]. Compared to similar tools like ProVerif [8] or AVISPA [38], Tamarin comes with a better user interface and more refined models for some primitives like Diffie-Hellman and XOR.

Modelling protocols. At its core, Tamarin is based on multiset rewriting. This means a protocol is represented using a series of multiset rewriting rules. A rule essentially dictates the labelled transition from one set of facts to another.

```
rule Resp: [ Resp1(idA), In(x) ] −[ Label(idA,x) ]→ [ Resp2(idA, x), Out(h(x)) ]
```

Example 1. Tamarin rewriting rule

A Tamarin rule is composed of four elements, namely its *name*, the set of *facts* that are input to the rule, the set of *labels* that are produced by the rule, and the set of facts that are output by the rule. In Example 1, if there exists a fact Resp1(idA) and there is an input message x in Tamarin's state, applying this rule will consume the fact Resp1, and produce the fact Resp2. The label Label is generated by the application of this rule. Out(...) is a special fact that represents the emission of a message over a public channel. In(...) is also a special fact that denotes the reception of a message. In this rule, idA and x are variables that can *a priori* be terms of any form or type.

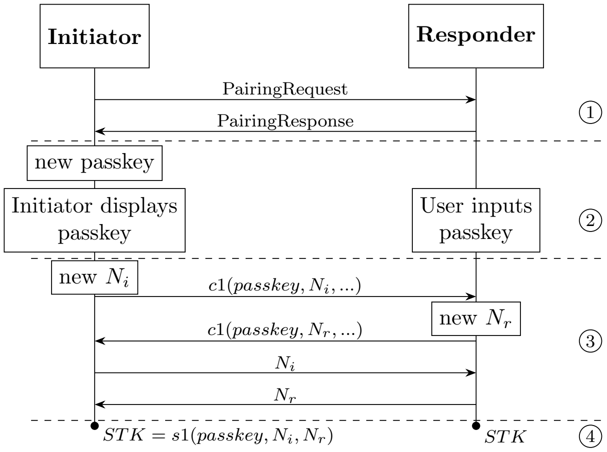

Fig. 2. BLE Legacy Passkey Entry

Modelling attacker. Tamarin analyses protocols in the Dolev-Yao model [20] where the attacker has full control over the communication channel: it is able to receive, intercept, modify, and forge messages. Tamarin automatically generates rules for the attacker, which enables it to perform common operations, like splitting and concatenating messages, etc. The attacker's knowledge is updated with each message sent on the public channel, hence with each Out(...) produced. Similarly, each message known to the attacker can be sent over the public channel, hence received in any In(...) fact.

In order to represent cryptographic operations, Tamarin enables to define function symbols and their relations through equations. It comes with existing symbols such as XOR, symmetric encryption, Diffie-Hellman, etc. The set of equations that relate functions together is called an *equational theory.*

Modelling properties. To gain insight and knowledge about protocols, Tamarin allows encoding mathematical properties, called *lemmas.* They are expressed using labels that are produced by rewriting rules.

lemma InitKeySecrecy:
"∀ id, stk #i. InitEndPairing(id, stk) @#i $\implies$ ∄ #j . K(stk) @#j"

Example 2. Tamarin lemma

Example 2 expresses a simple weak secrecy claim: if an Initiator ends the protocol with a certain key stk at time #i, the attacker is unable to retrieve it at any point in time. The lemmas are expressed as logical formulas, using quantifiers and negations, and the attacker knowledge is represented with fact K.

When provided with a lemma, Tamarin tries to prove it is true in all cases or provide an execution trace. This execution trace illustrates the different rules that are applied and the actions the attacker took to contradict the lemma.

From this trace, it is possible to manually identify the messages and computations an attacker does to invalidate the property studied. An other possibility is that Tamarin may not finish the proof within the allocated resources (time, memory). When Tamarin does not finish, it is possible to use an interactive mode and to prove the property manually by guiding Tamarin about the states to explore. Because Tamarin is, at its core, a prover, it does not yield all counterexamples of a lemma for a model. This means that when knowingly studying a flawed protocol, Tamarin is not able to enumerate all the attacks on this protocol. Furthermore, by default Tamarin considers cryptographic primitives to be perfect. However, some primitives have known weaknesses and some protocols use primitives in an incorrect way. Representing cryptographic imperfections requires an extra modelling step so Tamarin can include them in the model.

2.3 Related Work

Bluetooth technologies have been subject to many attacks over the years. A survey of those affecting BLE can be found in [10]. Some studies have focused on the security of the reconnection step: BIAS [1] considers the authentication protocol during reconnection in BR/EDR, KNOB [4] the key size reduction in BR/EDR, and BLESA [39] the reconnection in BLE.

There are also passive attacks on Bluetooth technologies. In BR/EDR, Legacy Pairing is vulnerable to offline key recovery from a capture of exchanged messages [32]. Legacy Pairing in BLE has the same flaw although the details differ [31]. In a Secure Pairing protocol, Lindell showed the possibility to retrieve passively an authentication secret [25], which applies to BLE and BR/EDR.

Rosa [30] proposed an active attack on Legacy Pairing in BLE that relies on a flawed cryptographic primitive. Researchers studied the use of ECDH in the Pairing protocols [7, 18], found flaws in the authentication of public keys and discussed possible attacks. Key size reduction is also studied in BLE [2], which proved to be vulnerable to some extent.

BlueMirror [15] proposed an extensive study of reflection attacks in Bluetooth technologies and showed their applicability to all of them. In [37], the authors define the concept of *pairing confusion*, where the attacker forces two devices to use two different Pairing protocols. In their attack, an attacker forces device A to complete Secure Passkey Entry while device B completes Secure Numeric Comparison. They show that in this setup, implementations do not allow the user to distinguish between both protocols. As a result, the attacker can complete them and retrieve the encryption key derived by each device.

Bluetooth was also studied from a formal perspective. Some studies performed manual proofs of some parts of Bluetooth. In [26], a proof of Secure Numeric Comparison is done. A formal analysis of Secure Passkey Entry is proposed in [36]. The security of the reconnection step in BR/EDR and BLE is studied in [21]. Formal studies using automated tools are also detailed in [12], [13], [29], [18], [23], [40] and [33]. They are discussed in depth in Section 4.3.

3 Formal models

This section details the choices made to model Bluetooth key agreements. We list in Section 3.2 all the cryptographic weaknesses that the attacker can exploit and explain how they are modelled in Tamarin. Then, the approach taken for modelling Bluetooth key agreements is presented.

3.1 Security model

Security properties. We study three kinds of security goals that are defined in the specification: confidentiality, authentication, and MitM protection.

Confidentiality and authentication are defined in this paper similarly to what is done in other Bluetooth formal analyses [33, 40]. For each Bluetooth technology, confidentiality comes from the secrecy of the keys derived at the end of the key agreement. Secrecy is modelled per participant, that is the secrecy of the keys derived by each participant is verified. To model authentication, we use the definition of *non-injective agreement* from Lowe's [27] taxonomy. Again, this property is modelled per participant, to ensure that no device has unknowingly completed a key agreement with an attacker.

Finally, the MitM protection [9](Vol 1, Part A, §5.2.3) is formalized. It represents the fact that an attacker should not be able to complete a key agreement with both participants at the same time and yet know the keys derived by each side. This property is also studied in [33] for BLE Secure Pairing.

Attacker model. There are three kinds of communication channels that are used in Bluetooth specifications. The first channel is the Bluetooth channel, which carries Bluetooth messages over the radio between devices. It is considered that the attacker has Dolev-Yao capabilities over the channel. The attacker is able to forge, modify, block, and relay messages over the radio.

The second kind of channel is the one used to model user interactions, because the user needs to perform some actions to complete most key agreements. In this model, the user is considered honest and performs actions as required by the specification. The attacker is supposed to have no access to the output/input of legitimate devices. When two devices output an information, the user verifies they match and confirms to continue the key agreement. When two devices expect an input, the user chooses a random number and fills it on both devices. When one device outputs an information and the other expects an input, the user enters the output information on the other device.

The third kind of channel is the OOB channel, that is used to transport information between two devices. This OOB channel is unspecified, but it is assumed that the attacker has no access to this channel given that this would break the security of the OOB protocols.

3.2 Representing cryptographic imperfections

By default, Tamarin assumes that cryptography is perfect, but primitives used in Bluetooth are known to have some weaknesses. This paragraph details how these imperfections are modelled in Tamarin.

Brute-force of low-entropy secrets. Some protocols rely on low-entropy secrets, which can be brute-forced by an attacker. Tis kind of vulnerability has various shapes depending on the technology and key agreement [2,15,25,31,32].

In Tamarin, the names used to represent nonces/passwords are unguessable by default: if there is a generated value *secret* and the attacker has access to h (*secret*), without further rule the attacker is unable to retrieve the value of *secret*. While this assumption is correct for some protocols (*e.g.*, if the secret value is 128-bit long), Bluetooth uses several low-entropy secrets that can be brute-forced in a practical time. To model this capability, special rules are created to output the targeted secret when the attacker has provided enough information.

```
rule Oracle_f4:
    [ LowEntropyf4(pk1, pk2, n, s), In(pk1), In(pk2), In(n), In(f4(pk1, pk2, n, s)) ]
  −[ AttackerRecoveredPasskey(s) ]→ [ Out(s) ]
```

Example 3. Oracle rule in Tamarin

The implementation of the passkey recovery [25] from BLE Secure Passkey Entry protocol is done with the rule depicted in Example 1. The function f4 is defined in the specification and is common to several Pairing methods. The methods that use a low-entropy secret generate the fact LowEntropyf4 $(pk1, pk2, n, s)$ that allows to enter this rule. The attacker also needs to prove knowledge of all the elements by sending them on the public channel. When used, this rule outputs the secret. The use of an explicit "oracle" rule makes it appear in Tamarin's execution traces, therefore one may follow easily whether such a rule occurs in a Tamarin attack trace. The ability of the attacker to brute-force downgraded keys, discussed in [2,33] is also modelled using such an oracle.

Malleable Commitment. This issue is present in BLE Legacy Pairing [30] and in BM Provisioning [15]. While both instances of commitment functions in Bluetooth have different cryptographic details, they are conceptually very similar. In BLE, the commitment protocol is displayed in step ③ of Figure 2: both devices exchange a commitment value computed from a key, a nonce, an authentication secret, and additional data. Device A sends the first commitment, followed by B. Then both devices exchange their nonces.

The vulnerabilities rely on the attacker posing as device B. Upon reception of A's commitment, the attacker replies to A with an arbitrary value. Then, A sends its nonce. From A's nonce and commitment, the attacker is able to recover an authentication secret. The attacker then crafts a nonce from the sent commitment and recovered authentication secret.

```
functions:
    aes_cmac/2, // Representation of cmac
    get_b1/3, // Used to retrieve first block
equations:
    get_b1(aes_cmac(k, <b1, b2>), k, b2) = b1,
    aes_cmac(k, <get_b1(c, k, b2), b2>) = c.
```

Example 4. Representing malleability in Tamarin

To implement the *malleable commitment* weakness, a specific equational theory is used. In Example 2, one can see the implementation for BM. In particular, it is necessary to define an equation to craft a nonce, represented here with get_b1. Then, one has to explicitly state that a confirmation that is used in this way is equal to a proper aes_cmac term. With this representation, Tamarin is able to find this class of attacks on the studied protocols.

This type of cryptographic problem strongly depends on the underlying cryptographic specification, and those equations are not suitable for all protocols. In Tamarin, it is impossible to state that this equation holds only if $b1$ and $b2$ have a specific size. As a result, those equations give the attacker more power than it has in practice and are not a generic representation of this kind of problem.

Small subgroup attack on ECDH implementation. In Bluetooth, incorrect ECDH implementations have led to some attacks on implementations [7,18]. This attack is a type of small subgroup attack that affects BR/EDR and BLE when the validity of received public keys is not verified. The representation of this type of attacks and more generally of incorrect implementations of the Diffie-Hellman protocol with Tamarin is extensively discussed in [18]. The authors provide a model of Secure Numeric Comparison with their representation.

In all Bluetooth technologies, the elliptic curves used are P-192 or/and P-256, which are defined over a field of prime order. Therefore, we adapted the representation of ECDH provided in their model to all Bluetooth technologies. Basically, each public key is represented as a group identifier, the neutral element of the group and the group element. When deriving a Diffie-Hellman key, if the attacker has managed to modify the group of an element, the key is considered leaked to the attacker. This is representative of elliptic curve cryptography on the groups used in Bluetooth, because an appropriate modification of a public key yields a Diffie-Hellman secret that is on a group of low order (as low as 2). In that case, the secret becomes easily retrievable using brute-force.

3.3 Modelling Bluetooth key agreement protocols

When modelling key agreements in Bluetooth, one needs to tackle the diversity of protocols. In order to model them accurately, one needs to model the user interaction required to complete each of them. In the specification, a single protocol may have several user interaction variations, depending on the input/output capabilities of both devices. For example, in BLE Legacy Passkey Entry, a device may have an input, an output or both. Whether the device outputs or waits for a numeric code depends on the other device's input-output capabilities. To address this variation, Legacy Passkey Entry is modelled as three sub-protocols to represent the different user interactions required. This also applies to other Pairing protocols, and increases the number of protocols that are represented. In total, there are 13 BLE protocols, 11 BR/EDR protocols, and 8 BM protocols to consider all the identified variations.

In practice, the choice of the protocol to use between two legitimate devices is done in the very first step, which is the Feature Exchange. An active attacker has the ability to modify the features sent by each device, and therefore the ability to force the protocol used on each side of the connection. Therefore, studying each pair of protocols makes sense from a Bluetooth's point of view. Studying the interaction of all possible pairs of protocols for each technology requires studying 354 ($13^2 + 11^2 + 8^2$) distinct cases, each case containing several properties to analyse. This forms the baseline of the models presented in this paper.

In total, there is one model per technology, containing all sub-protocols identified for this technology. Their respective size is detailed in Table 2. Although the models are large, the analysis of all lemmas of all protocols is efficient. The analysed configurations completed in less than 77 hours of CPU time.

Table 2. Sizes of the Tamarin models

Model	# rules	# restrictions	# macros	# lemmas	# lines
BR/EDR	117	13	165	605	∼11000
BLE	123	12	220	845	∼14400
BM	57	8	100	640	∼6600

Using the models. The Tamarin preprocessor is used to prevent Tamarin from processing parts of the models that are irrelevant for an interaction. For this study, the use of macros yielded a speedup of two to three orders of magnitude for Tamarin. As a result all interactions can be studied in practical time.

Moreover, to gain more insight into the strengths and weaknesses of each protocol, one may want to study the effects of specific imperfections. Similarly, to study the effects of a patch, one may want to study the impact if only one of the two devices is patched. For example, in [18] the authors analyse the outcome of having one device with a patched ECDH implementation and another with a flawed one. The proposed models support this type of configuration. For example, it is possible to study all the mentioned protocols while preventing the attacker to brute-force low-entropy secrets using a simple command-line flag. Likewise, it is possible to study all the relevant protocols where one device has a patched version of ECDH using another flag. Overall, there are different flag combinations that allow to study different configurations of a model, from the same source file.

4 Security analysis

The results of our study are presented in this section, and then compared with existing results from the literature.

4.1 Analysis of the results

Various configurations of imperfections and patches are analysed. Each study requires to run Tamarin on the models to try to prove all lemmas, yielding a proof or an attack trace for each lemma. For example, when ECDH problems are patched and devices are not vulnerable to keysize reduction, Tamarin identifies 659 attack traces. Each attack trace is manually analysed to identify to which result it is linked to. Complete annotated result tables are released along with the models, this section only displays a synthesis of the results.

Table 3. Attacks identified by presented formal models on Bluetooth key agreements

Label	Attack	Technology			This paper	Wu et al. [40]	Cremers et al. [18]	Jangid et al. [23]	Shi et al. [33]
		BR/EDR	BLE	BM					
A1	Reflection attack on Legacy PIN Pairing	×			✓				
A2	Brute-force PIN from protocol	×			✓				
A3	JustWorks is not authenticated	×	×		✓	✓			
A4	Pairing Method confusion	×	×		✓	✓		✓	✓
A5	Reflection attack in Secure Passkey Entry	×	×		✓			✓	
A6	**(new)** Extenstion to Pairing Method confusion	×			✓				
A7	**(new)** Pairing Mode confusion	×			✓				
A8	Invalid Curve attack	×	×		✓		✓		
A9	Reflection attack in Legacy Pairing		×		✓				
A10	Brute-force passkey from protocol		×		✓				
A11	Malleable commitment in Legacy Passkey Entry		×		✓				
A12	**(new)** Extension to Pairing Method confusion		×		✓				
A13	**(new)** Pairing Mode confusion		×		✓				
A14	Keysize downgrade in BLE-SC		×		✓				✓
A15	OOBno is not authenticated			×	✓	✓			
A16	Reflection attack in Provisioning			×	✓	✓			
A17	Brute-force AuthData from protocol			×	✓				
A18	**(new)** Lack of key confirmation in Provisioning			×	✓				
A19	(combination) Reflection and AuthData brute-force			×	✓				
A20	(combination) Reflection and AuthData retrieval			×	✓				
A21	(combination) AuthData retrieval and malleable commitment			×	✓				

Table 3 summarizes the attacks identified by the presented models. Moreover, attacks relying on different core assumptions, like semi-compromised devices, are not displayed. Most attacks were discovered across the years through manual analysis, and are accurately picked up by our Tamarin models. We only detail below the new attacks obtained.

In BM, we identify a lack of key confirmation at the end of the protocol (**A18**): an attacker can prevent a new device from joining the network, while making the network believe that the device has successfully joined. This leads to a Denial of Service (DoS), which exact effects are implementation-dependent.

Several protocol confusions attacks are picked up. The original attack [37] (**A4**), describes a confusion between Secure Passkey Entry and Secure Numeric

Comparison that affects BR/EDR and BLE. Tamarin identifies four novel confusion attacks for other pairs of protocols, that break all studied security properties.

- **A6**: Legacy PIN Pairing / Secure Numeric Comparison (BR/EDR)
- **A7**: Legacy PIN Pairing / Secure Passkey Entry (BR/EDR)
- **A12**: Legacy Passkey Entry / Secure Numeric Comparison (BLE)
- **A13**: Legacy Passkey Entry / Secure Passkey Entry (BLE)

The original attack is a Pairing confusion regarding the method, whereas the new ones are Pairing confusions regarding the mode. More importantly, the original attack, as well as attacks **A6** and **A12** can be mitigated by improving the display of expected user actions. In Numeric Comparison, the expected action is for the user to confirm that two numeric codes are equal, while for Passkey Entry the expected action is that the user inputs a numeric code displayed by one device on the other. Some implementations do not have a correct display of expected user actions, which leads to the possible confusion: users input the confirmation code into another device [37].

By contrast, attacks **A7** and **A13** bypass this mitigation because all involved protocols have identical user actions. They have been attributed CVE identifiers by the Bluetooth SIG and are described in more details below. Both attacks share a similar setup, but rely on different cryptographic weaknesses. The attacker forces one device to use a Legacy protocol which has the same user interaction as Secure Passkey Entry. The attacker uses a cryptographic issue to complete the Legacy protocol, retrieving the encryption key and the passkey/PIN used. Then, the attacker uses the gained knowledge of the passkey to complete the Secure Passkey Entry protocol.

Attack A7: Pairing Mode Confusion in BR/EDR - CVE-2022-25837. The attack is depicted in Figure 3. The attacker forces the Initiator to use the Secure Passkey Entry protocol and the Responder to use the PIN Pairing protocol. To do so, the attacker sends the first message of the PIN Pairing protocol to the Responder which forces it to use this protocol. Then, upon connection of the Initiator, the attacker announces support for Secure Pairing in its features. By modifying its input-output capabilities, the attacker forces a valid user interaction between PIN Pairing and Secure Passkey Entry, for example the Initiator may display a numeric code (the passkey) and the Responder asks the user to input a numeric code (the PIN). The PIN can be recovered from the values exchanged in the PIN Pairing protocol and the authentication protocol which serves as key confirmation [32]. Because the PIN is the passkey in the Secure Passkey Entry protocol, the attacker completes the key agreement with the Initiator. In the end, the attacker has successfully completed Pairing with both devices and shares a different encryption key with each of them.

Attack A13: Pairing Mode Confusion in BLE - CVE-2022-25836. The attack is depicted in Figure 4. Function $c1$ is defined in the specification, function `get_n` computes a correct nonce given a confirmation value. This results in the malleability of the commitment function in Legacy Passkey Entry protocol, as

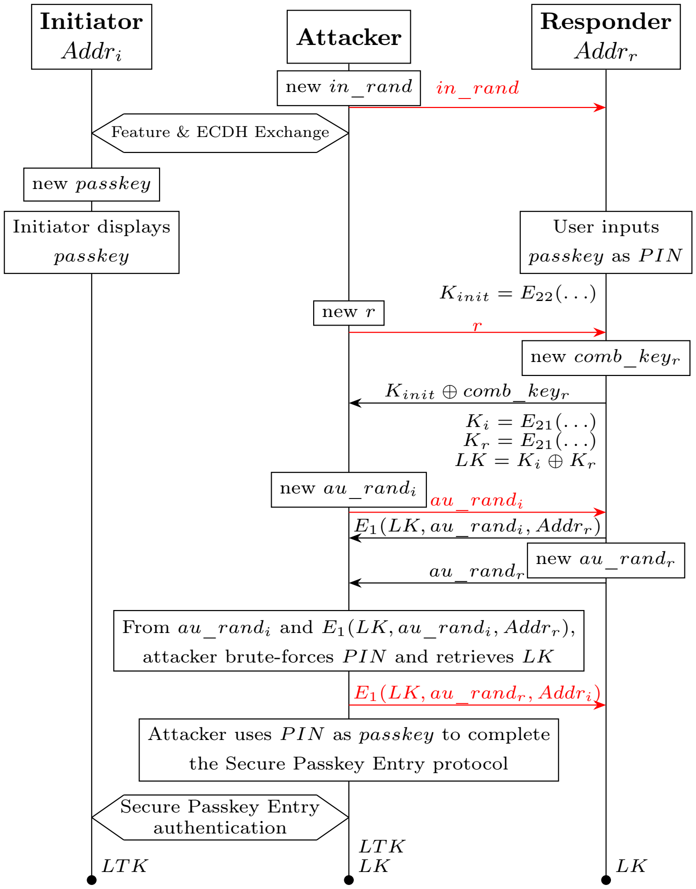
Initiator
$Addr_i$
Attacker
Responder
$Addr_r$
new in_rand
in_rand
Feature & ECDH Exchange
new $passkey$
Initiator displays $passkey$
User inputs $passkey$ as PIN
$K_{init} = E_{22}(\ldots)$
new r
r
new $comb\_key_r$
$K_{init} \oplus comb\_key_r$
$K_i = E_{21}(\ldots)$
$K_r = E_{21}(\ldots)$
$LK = K_i \oplus K_r$
new au_rand_i
au_rand_i
$E_1(LK, au\_rand_i, Addr_r)$
new au_rand_r
au_rand_r
From au_rand_i and $E_1(LK, au\_rand_i, Addr_r)$, attacker brute-forces PIN and retrieves LK
$E_1(LK, au\_rand_r, Addr_i)$
Attacker uses PIN as $passkey$ to complete the Secure Passkey Entry protocol
Secure Passkey Entry authentication
LTK
LTK
LK
LK

Fig. 3. Pairing Mode Confusion in BR/EDR **A7**

found by Rosa [30]. The attacker can force the Initiator to use the Legacy Passkey Entry protocol and the Responder to use the Secure Passkey Entry protocol by modifying the input-output capabilities and the *Secure* flag during Feature Exchange. The attacker then completes the protocol on the Legacy side, which makes use of the ability to brute-force the passkey and of the malleability of the commitment in Legacy Pairing. This enables the attacker to recover the passkey, thus to have a legitimate Secure Passkey Entry interaction with the Responder. In the end, the attacker has completed Pairing with both devices while sharing a different encryption key with each of them.

4.2 Practical implementation

To assess their applicability, Pairing Mode confusion attacks have been tested on off-the-shelf devices. In BR/EDR and BLE, the specification defines a complete

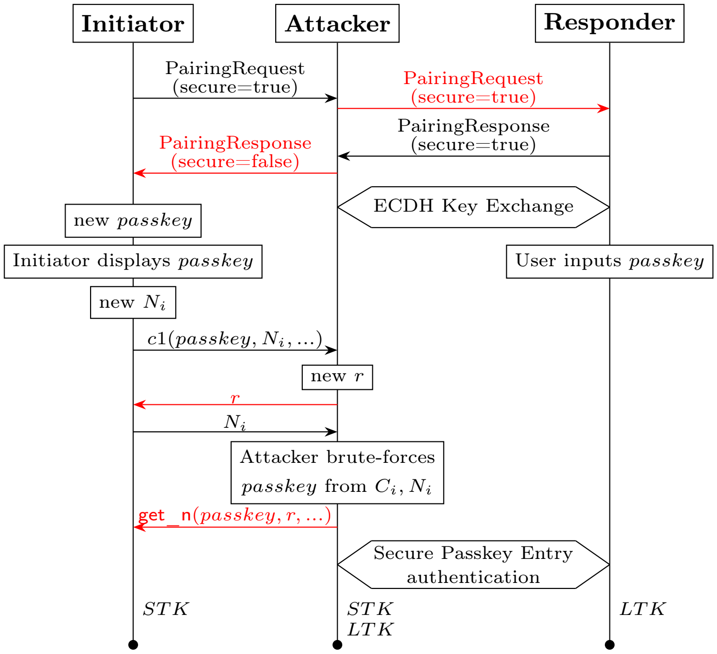
Initiator
Attacker
Responder
PairingRequest (secure=true)
PairingRequest (secure=true)
PairingResponse (secure=true)
PairingResponse (secure=false)
ECDH Key Exchange
new *passkey*
Initiator displays *passkey*
User inputs *passkey*
new N_i
$c1(passkey, N_i, ...)$
new r
r
N_i
Attacker brute-forces *passkey* from C_i, N_i
get_n$(passkey, r, ...)$
Secure Passkey Entry authentication
STK
STK
LTK
LTK

Fig. 4. Pairing Mode Confusion in BLE **A13**

protocol stack, from the physical layer to the application layer. Pairing happens in the intermediate layers of the protocol stack. Both Pairing Mode confusions require the attacker to implement a custom Pairing procedure. Hence, to perform the attack one needs the ability to receive and craft Pairing messages.

To implement the attack in BR/EDR, the research-oriented firmware BrakTooth [22] is used. It is to be noted that the core feature necessary to implement this attack, namely message injection in the layer handling Pairing messages in BR/EDR, is an undocumented component of the firmware. Thus, a custom driver is developed to create a MitM framework out of two dongles flashed with this firmware. Then, the handling of Pairing messages is reimplemented to implement both sides of the attacks. To the best of the authors knowledge, this is the first practical implementation of BR/EDR MitM on a Pairing protocol.

To implement the attack in BLE, the framework Mirage [11] that has built-in support of BLE MitM is used. As with BR/EDR, the handling of Pairing messages is reimplemented to implement both sides of the attacks.

For each technology, two Android phones are used as targets. The attack is successful in both cases, meaning that the attacker is able to retrieve the encryption key with both devices. It is important to raise that the user interaction is the same for both the Legacy and the Secure modes. Finally, it is noted that the user interaction on Android is identical between BR/EDR and BLE. Though it was not tested, it could be used to create Pairing Technology confusion attacks, by using a different technology with each target.

4.3 Related Work

There are few published formal symbolic analyses of the Bluetooth protocols involving automated tools. For completeness, it is noted that [13] performed a ProVerif [8] analysis of Numeric Comparison but did not identify any weakness. In [12] the authors demonstrated that injective key agreement does not hold in Numeric Comparison. A study of misbinding attacks is performed in [29] using ProVerif. All those studies focus on various definitions of authentication for one or two Pairing protocols, while the present paper considers all Bluetooth key agreements. The relevance of our model and results are discussed with respect to more accurate models of Bluetooth key agreements: [18], [40], [23] and [33].

In [18], the authors use Tamarin to study the security of the Secure Numeric Comparison protocol with regards to small subgroup attacks on the Diffie-Hellman key exchange, extending results from [7]. In the present study, the analysis of BR/EDR and BLE is also done considering two, one or none of the devices patched. This allows identifying more possible attack scenarios where some attacks are combined. The results for those configurations are not reported in this paper due to size constraints, but the models are available for further study.

In [40], the authors also study the Pairing protocols in BR/EDR and BLE. However, they do not take into account Legacy protocols, and do not perform a systematic study of possible confusion attacks. Also, their model of Secure protocols considers perfect cryptographic primitives, this makes them miss attacks on ECDH and the reflection attack on Secure Passkey Entry, which are correctly identified by our models. In Bluetooth Mesh, the authors propose a patch for the reflection attack identified (**A15**). However, those patches were analysed with our models and proven insecure, as they still allow an attacker to compromise communications. The attacks on the patches rely on a known weakness of Bluetooth Mesh in the use of a malleable commitment function based on AES-CMAC. As a result, the attacks on the patches are similar to the attacks on the original Provisioning protocol described in [15].

In [23], the authors analyse Secure Passkey Entry in Tamarin. Among the attacks they identified, there are Pairing Confusion [37] and the reflection attack [15]. These attacks were known before and also retrieved by our analysis. The other attacks they identified rely on the hypothesis that the attacker gains the passkey in other ways, due to implementation problems (*e.g.*, bad randomness). In our model we decided not to make any implementation-related assumptions, meaning that we do not catch these attacks. Furthermore, their study tackles only one Pairing protocol, while ours encompasses all Bluetooth key agreements and considers more cryptographic imperfections.

The authors of [33] study protocol confusion, but only for BLE Secure Pairing. They did neither model BR/EDR Secure Pairing nor Legacy protocols. They also study the possibility of keysize downgrade in BLE Secure Pairing, but do not model any other cryptographic weakness. They identify another type of attack that may lead to a DoS called keysize confusion attack. The keysize downgrade is accurately picked up by our analysis, but the keysize confusion is not caught because DoS attacks are out of scope of this paper. It is worth noting that our work confirms that the keysize downgrade attack is valid in BLE Secure Pairing, but it shows that it does not affect BLE Legacy Pairing. Upon verification, the reason is that the bytes containing the key size are part of the authentication protocol in all Legacy Pairing protocols, but are not in any Secure Pairing protocol. As a result, an attacker can modify the keysize bytes without affecting the protocol in BLE Secure Pairing, but cannot do so in BLE Legacy Pairing.

5 Conclusion

Bluetooth has a security mode in BLE and BR/EDR that forces connections to use Secure Pairing modes only and 128-bit keys. For example, this mode can be used for critical applications. Whether it is implemented and enforced remains an implementation and configuration matter.

The attacks presented in this paper demonstrate that the knowledge of the configuration of one of the two devices is not enough to have complete security guarantees. If one device is configured to use only Secure Pairing but the peer device still allows Legacy Pairing, then the communication between them is not immune to attacks. Moreover, the user is not able to detect the attack because the mode confusion keeps an identical user interaction as a legitimate exchange.

In its statement about the original Pairing Method confusion from [37], the Bluetooth SIG [34] recommends device manufacturers to make it more obvious which interaction is expected from users, to avoid confusions. They did not modify the underlying protocols, hence no patch is enforced for this problem. The confusions presented in this paper bypass this mitigation because the user interaction is not only similar but identical for both protocols.

For the Pairing Mode confusion, weaknesses in Legacy modes are used to break a Secure mode. Because Legacy Pairing protocols are structurally broken, they cannot be patched while remaining compatible with older devices. Security-wise, the only definitive technical solution consists in removing Legacy Pairing from implementations and specifications to make devices compliant with Legacy Pairing gradually disappearing. However, removing Legacy Pairing would prevent communication with devices that do not support Secure Pairing.

In its statements about those vulnerabilities, the Bluetooth SIG recommends to disable Legacy Pairing and to implement better user interaction to indicate if Legacy mode is being used. However, this is not always possible, as not all devices possess a screen to accurately inform the user. Overall, for an informed user, the best way to remain protected from such attacks is to verify that both communicating devices are up to date and have disabled Legacy Pairing.

Acknowledgements. We kindly thank the authors of [40] for their insightful remarks about Bluetooth formal modelling and Bluetooth Mesh Provisioning.

References

1. Antonioli, D., Tippenhauer, N.O., Rasmussen, K.: BIAS: Bluetooth Impersonation AttackS. In: 41st IEEE Symposium on Security and Privacy, SP 2020, San Francisco, CA, USA, May 18-21, 2020. pp. 549–562. IEEE Computer Society (2020). https://doi.org/10.1109/SP40000.2020.00093
2. Antonioli, D., Tippenhauer, N.O., Rasmussen, K.: Key Negotiation Downgrade Attacks on Bluetooth and Bluetooth Low Energy. ACM Trans. Priv. Secur. **23**(3) (jul 2020). https://doi.org/10.1145/3394497
3. Antonioli, D., Tippenhauer, N.O., Rasmussen, K., Payer, M.: BLURtooth: Exploiting Cross-Transport Key Derivation in Bluetooth Classic and Bluetooth Low Energy. In: ASIA CCS '22: ACM Asia Conference on Computer and Communications Security, Nagasaki, Japan, 30 May 2022 - 3 June 2022. pp. 196–207. ACM (2022). https://doi.org/10.1145/3488932.3523258
4. Antonioli, D., Tippenhauer, N.O., Rasmussen, K.B.: The KNOB is Broken: Exploiting Low Entropy in the Encryption Key Negotiation Of Bluetooth BR/EDR. In: 28th USENIX Security Symposium, USENIX Security 2019, Santa Clara, CA, USA, August 14-16, 2019. pp. 1047–1061. USENIX Association (2019)
5. Basin, D.A., Sasse, R., Toro-Pozo, J.: The EMV standard: Break, Fix, Verify. In: 2021 IEEE Symposium on Security and Privacy (SP), May 23-27, 2021, San Francisco, CA, US. pp. 1766–1781. IEEE (May 2021). https://doi.org/10.1109/SP40001.2021.00037
6. Bhargavan, K., Blanchet, B., Kobeissi, N.: Verified Models and Reference Implementations for the TLS 1.3 Standard Candidate. In: 38th IEEE Symposium on Security and Privacy, SP 2017, San Jose, CA, USA, May 22-26, 2017. pp. 483–502. IEEE Computer Society (2017). https://doi.org/10.1109/SP.2017.26
7. Biham, E., Neumann, L.: Breaking the Bluetooth Pairing - The Fixed Coordinate Invalid Curve Attack. In: Selected Areas in Cryptography - SAC 2019 - 26th International Conference, Waterloo, ON, Canada, August 12-16, 2019. Lecture Notes in Computer Science, vol. 11959, pp. 250–273. Springer (2019). https://doi.org/10.1007/978-3-030-38471-5_11
8. Blanchet, B.: Automatic Verification of Security Protocols in the Symbolic Model: The Verifier ProVerif. In: Foundations of Security Analysis and Design VII - FOSAD 2012/2013 Tutorial Lectures. Lecture Notes in Computer Science, vol. 8604, pp. 54–87. Springer (2013). https://doi.org/10.1007/978-3-319-10082-1_3
9. Bluetooth SIG: Bluetooth Core Specification (01 2023), v5.4
10. Cäsar, M., Pawelke, T., Steffan, J., Terhorst, G.: A survey on Bluetooth Low Energy security and privacy. Comput. Networks **205** (2022). https://doi.org/10.1016/j.comnet.2021.108712
11. Cayre, R., Nicomette, V., Auriol, G., Alata, E., Kaâniche, M., Marconato, G.V.: Mirage: Towards a Metasploit-Like Framework for IoT. In: 30th IEEE International Symposium on Software Reliability Engineering, ISSRE 2019, Berlin, Germany, October 28-31, 2019. pp. 261–270. IEEE Computer Society (2019). https://doi.org/10.1109/ISSRE.2019.00034
12. Chang, R., Shmatikov, V.: Formal Analysis of Authentication in Bluetooth Device Pairing. In: Proceedings of the Joint Workshop on Foundations of Computer Security and Automated Reasoning for Security Protocol Analysis, FCS-ARSPA'07. pp. 45–62 (2007)

13. Chothia, T., Smyth, B., Staite, C.: Automatically Checking Commitment Protocols in ProVerif without False Attacks. In: Principles of Security and Trust - 4th International Conference, POST 2015, London, UK, April 11-18, 2015. Lecture Notes in Computer Science, vol. 9036, pp. 137–155. Springer (2015). https://doi.org/10.1007/978-3-662-46666-7_8
14. Claverie, T., Avoine, G., Delaune, S., Esteves, J.L.: Tamarin-based Analysis of Bluetooth Uncovers Two Practical Pairing Confusion Attacks, https://hal.science/hal-04079883
15. Claverie, T., Lopes-Esteves, J.: BlueMirror: Reflections on Bluetooth Pairing and Provisioning Protocols. In: 15th IEEE Workshop on Offensive Technologies, WOOT 2021, San Francisco, CA, USA, May 27, 2021. pp. 339–351. IEEE Computer Society (2021). https://doi.org/10.1109/SPW53761.2021.00054
16. Cremers, C., Dehnel-Wild, M.: Component-Based Formal Analysis of 5G-AKA: Channel Assumptions and Session Confusion. In: 26th Annual Network and Distributed System Security Symposium, NDSS 2019, San Diego, California, USA, February 24-27, 2019. The Internet Society (2019)
17. Cremers, C., Horvat, M., Scott, S., van der Merwe, T.: Automated Analysis and Verification of TLS 1.3: 0-RTT, Resumption and Delayed Authentication. In: 37th IEEE Symposium on Security and Privacy, SP 2016, San Jose, CA, USA, May 22-26, 2016. pp. 470–485. IEEE Computer Society (2016). https://doi.org/10.1109/SP.2016.35
18. Cremers, C., Jackson, D.: Prime, Order Please! Revisiting Small Subgroup and Invalid Curve Attacks on Protocols using Diffie-Hellman. In: 32nd IEEE Computer Security Foundations Symposium, CSF 2019, Hoboken, NJ, USA, June 25-28, 2019. pp. 78–93. IEEE Computer Society (2019). https://doi.org/10.1109/CSF.2019.00013
19. Cremers, C., Kiesl, B., Medinger, N.: A Formal Analysis of IEEE 802.11's WPA2: Countering the Kracks Caused by Cracking the Counters. In: 29th USENIX Security Symposium, USENIX Security 2020, August 12-14, 2020. pp. 1–17. USENIX Association (2020)
20. Dolev, D., Yao, A.C.: On the security of public key protocols. IEEE Trans. Inf. Theory **29**(2), 198–207 (1983). https://doi.org/10.1109/TIT.1983.1056650
21. Fischlin, M., Sanina, O.: Cryptographic Analysis of the Bluetooth Secure Connection Protocol Suite. In: Advances in Cryptology - ASIACRYPT 2021 - 27th International Conference on the Theory and Application of Cryptology and Information Security, Singapore, December 6-10, 2021, Proceedings, Part II. Lecture Notes in Computer Science, vol. 13091, pp. 696–725. Springer (2021). https://doi.org/10.1007/978-3-030-92075-3_24
22. Garbelini, M.E., Bedi, V., Chattopadhyay, S., Sun, S., Kurniawan, E.: BrakTooth: Causing Havoc on Bluetooth Link Manager via Directed Fuzzing. In: 31st USENIX Security Symposium, USENIX Security 2022, Boston, MA, USA, August 10-12, 2022. pp. 1025–1042. USENIX Association (2022)
23. Jangid, M.K., Zhang, Y., Lin, Z.: Extrapolating Formal Analysis to Uncover Attacks in Bluetooth Passkey Entry Pairing. In: 30th Annual Network and Distributed System Security Symposium, NDSS 2023, San Diego, California, USA, February 27 - March 3, 2023. The Internet Society (2023)
24. Jason, M.: New Wireless Trends and Forecasts for the Next 5 Years. https://www.bluetooth.com/blog/new-trends-and-forecasts-for-the-next-5-years/
25. Lindell, A.Y.: Attacks on the Pairing Protocol of Bluetooth v2.1. https://www.blackhat.com/presentations/bh-usa-08/Lindell/BH_US_08_Lindell_Bluetooth_2.1_New_Vulnerabilities.pdf (June 2008), BlackHat USA

26. Lindell, A.Y.: Comparison-Based Key Exchange and the Security of the Numeric Comparison Mode in Bluetooth v2.1. In: Topics in Cryptology – CT-RSA 2009, vol. 5473, pp. 66–83. Springer (2009)
27. Lowe, G.: A Hierarchy of Authentication Specification. In: Computer Security Foundations Workshop 1997. IEEE Computer Society (1997). https://doi.org/10.1109/CSFW.1997.596782
28. Meier, S., Schmidt, B., Cremers, C., Basin, D.A.: The TAMARIN Prover for the Symbolic Analysis of Security Protocols. In: Computer Aided Verification - 25th International Conference, CAV 2013, Saint Petersburg, Russia, July 13-19, 2013. Proceedings. Lecture Notes in Computer Science, vol. 8044, pp. 696–701. Springer (2013). https://doi.org/10.1007/978-3-642-39799-8_48
29. Peltonen, A., Sethi, M., Aura, T.: Formal verification of misbinding attacks on secure device pairing and bootstrapping. Journal of Information Security and Applications (Apr 2020). https://doi.org/10.1016/j.jisa.2020.102461
30. Rosa, T.: Bypassing Passkey Authentication in Bluetooth Low Energy. IACR Cryptol. ePrint Arch. p. 309 (2013), http://eprint.iacr.org/2013/309
31. Ryan, M.: Bluetooth: With Low Energy Comes Low Security. In: 7th USENIX Workshop on Offensive Technologies, WOOT 2013, Washington, D.C., USA, August 13, 2013. USENIX Association (2013)
32. Shaked, Y., Wool, A.: Cracking the Bluetooth PIN. In: Proceedings of the 3rd International Conference on Mobile Systems, Applications, and Services, MobiSys 2005, Seattle, Washington, USA, June 6-8, 2005. pp. 39–50. ACM (2005). https://doi.org/10.1145/1067170.1067176
33. Shi, M., Chen, J., He, K., Zhao, H., Jia, M., Du, R.: Formal Analysis and Patching of BLE-SC Pairing. In: 32nd USENIX Security Symposium, USENIX Security 2023, Anaheim CA, USA, August 9-11, 2023. USENIX Association (2023)
34. SIG, B.: Bluetooth SIG Statement Regarding the Method-Confusion Pairing Vulnerability. https://www.bluetooth.com/learn-about-bluetooth/key-attributes/bluetooth-security/method-vulnerability/
35. Tillmanns, J., Classen, J., Rohrbach, F., Hollick, M.: Firmware Insider: Bluetooth Randomness is Mostly Random. In: 14th USENIX Workshop on Offensive Technologies, WOOT 2020, August 11, 2020. USENIX Association (2020), https://www.usenix.org/conference/woot20/presentation/tillmanns
36. Troncoso, M., Hale, B.: The Bluetooth CYBORG: Analysis of the Full Human-Machine Passkey Entry AKE Protocol. In: 28th Annual Network and Distributed System Security Symposium, NDSS 2021, February 21-25, 2021. The Internet Society (2021)
37. von Tschirschnitz, M., Peuckert, L., Franzen, F., Grossklags, J.: Method Confusion Attack on Bluetooth Pairing. In: 42nd IEEE Symposium on Security and Privacy, SP 2021, San Francisco, CA, USA, 24-27 May 2021. pp. 1332–1347. IEEE Computer Society (2021). https://doi.org/10.1109/SP40001.2021.00013
38. Viganò, L.: Automated Security Protocol Analysis With the AVISPA Tool. In: Annual Conference on Mathematical Foundations of Programming Semantics 2005. Elsevier (2005). https://doi.org/10.1016/j.entcs.2005.11.052
39. Wu, J., Nan, Y., Kumar, V., Tian, D.J., Bianchi, A., Payer, M., Xu, D.: BLESA: spoofing attacks against reconnections in bluetooth low energy. In: 14th USENIX Workshop on Offensive Technologies, WOOT 2020, August 11, 2020. USENIX Association (2020), https://www.usenix.org/conference/woot20/presentation/wu

40. Wu, J., Wu, R., Xu, D., Tian, D.J., Bianchi, A.: Formal Model-Driven Discovery of Bluetooth Protocol Design Vulnerabilities. In: 43rd IEEE Symposium on Security and Privacy, SP 2022, San Francisco, CA, USA, May 22-26, 2022. pp. 2285–2303. IEEE Computer Society (2022). https://doi.org/10.1109/SP46214.2022.9833777
41. Xu, F., Diao, W., Li, Z., Chen, J., Zhang, K.: BadBluetooth: Breaking Android Security Mechanisms via Malicious Bluetooth Peripherals. In: 26th Annual Network and Distributed System Security Symposium, NDSS 2019, San Diego, California, USA, February 24-27, 2019. The Internet Society (2019)
42. Zhang, Y., Weng, J., Dey, R., Jin, Y., Lin, Z., Fu, X.: Breaking Secure Pairing of Bluetooth Low Energy Using Downgrade Attacks. In: 29th USENIX Security Symposium, USENIX Security 2020. pp. 37–54. USENIX Association (2020), https://www.usenix.org/conference/usenixsecurity20/presentation/zhang-yue