Security Comparison: Bluetooth^{TM} Communications vs. 802.11

Thomas G. Xydis Ph.D.
Simon Blake-Wilson
Bluetooth Security Experts Group

11-10-2001
revised 5-15-2002

1.0 Introduction

Several attacks on IEEE 802.11b have been described in the media [1]. It has been shown that the WEP security framework used in IEEE 802.11 is susceptible to both attacks on data content and user authentication. These exposures allow an attacker to both inappropriately intercept data and also gain access to a network by impersonating a valid user.

Bluetooth and IEEE 802.11b are different, complementary technologies [10]. IEEE 802.11b is largely applied to LAN access, while Bluetooth LAN access is only one of many applications, most of which focus on smaller personal area networks (PANs). Different target applications and technology dictate different security architectures. With the differences between Bluetooth technology and IEEE 802.11b in mind, one may question the validity of comparing the security architectures of the two technologies. We feel, however, that such a comparison *is* valid. Indeed, from a user perspective the two technologies are really quite similar. Both are methods which allow computers to communicate to other devices, both use wireless technology, both operate in the 2.4 GHz spread spectrum band, etc. Due to these similarities, the public sometimes confuses Bluetooth communications with IEEE 802.11b. In addition, 802.11b security concerns have been unjustifiably applied to Bluetooth communications. However, these attacks do not apply to Bluetooth technology.

Here we discuss the two main attacks on 802.11b that have been described in the literature. We also explain why these attacks are not effective with Bluetooth wireless communications.

2.0 802.11b Eavesdropping

When a user sends data over a wireless network, he has a reasonable expectation that such data is not easily readable by unauthorized persons. Unlike a wired network, which requires a physical intrusion, wireless data packets can be received by anyone nearby with an appropriate receiver, potentially outside of the physical security barriers of an organization. This allows, so called parking lot attacks, in which an attacker sits in a car in the parking lot of the intended victim. Accordingly, both Bluetooth and 802.11 technologies utilize data encryption in lower network layers.

The 802.11b specification utilizes a security framework called wireless equivalent privacy (WEP) protocol. A key component of WEP is the use of the stream cipher RC4. RC4 is a well-known and commonly used stream cipher, but its use in 802.11b is questionable owing to the nature of a wireless packet network.

RC4 operates by XORing the plaintext data with an encryption key stream. The result is commonly called ciphertext. RC4 is initialized with a secret WEP key and a public 24 bit long IV (initialization vector). If an attacker knows a plaintext and ciphertext pair *a-priori*, he can compute the encryption key stream using the XOR operation. Due to the low entropy of most plaintext messages, if an attacker can record a large number of ciphertext messages he can also compute the encryption key stream. For this reason users of RC4 are encouraged to change the encryption key on every message. The basic problem with RC4 over 802.11b is that wireless channels, will, by nature, occasionally lose data or drop packets. Thus, the synchronization between the encryptor and decryptor is difficult to maintain for any length of time. To overcome this limitation, WEP maintains synchronization by changing the 24-bit initialization vector (IV) on each packet. However 802.11b packets are relatively short. Therefore, one can expect a key/IV combination to be repeated every few seconds. Therefore one can expect a key/IV combination to be repeated relatively
frequently. For more details, see Walker [2], and Borisov et al [3].

Furthermore, there are more sophisticated attacks that exploit subtle properties of the key-scheduling algorithm of RC4 deployed with WEP. In July 2001, Fluhrer, Mantin, and Shamir [4] showed how to exploit the particular way that 802.11b derives the encryption key stream from the initialization vector IV and the shared key to completely recover the secret key. The paper shows how to recover the key, byte by byte, by exploiting a bias in the statistical distribution of the candidate secret keys that becomes manifest when observing the first byte of the key stream. By feeding the system with sufficiently many publicly known IV strings, the authors show how this bias can be exploited to correctly determine the key, one byte at a time. A team of AT&T researchers [2] soon after demonstrated a practical implementation of the attack using off-the-shelf equipment. The implementation showed that the key could be recovered by eavesdroppinging roughly 5 million encrypted packets. Thus WEP encryption is relatively easy to break.

3.0 802.11b False Authentication

To gain access to a network, a user must be authenticated. While authentication is typically done at a higher network level, 802.11b and Bluetooth technologies also support device authentication.

In 802.11b authentication is performed by a challenge response procedure using a shared secret. After requesting authentication, the authenticator sends the initiator a 128-octet random number challenge. The initiator encrypts the challenge using the shared secret and transmits it back to the authenticator. Encryption is performed by XORing the challenge with a pseudo-random string formed by the shared secret and a public IV. Note that the only thing that changes from authentication to authentication with a specific user is the plaintext message. [1]

A simple and powerful attack on this authentication mechanism is presented by Arbaugh, et. al [5]. First the intruder determines the pseudorandom string by recording the challenge (plaintext) and the response (ciphertext) and XORing them. He then impersonates the victim by using the pseudorandom string to compute the response to subsequent challenges. Notice that the attacker never needs to determine the shared secret; knowledge of the pseudorandom string is sufficient.

4.0 Device Authentication in Bluetooth Technology

Like 802.11b, Bluetooth technology provides a method for authenticating devices. Device authentication is provided using a shared secret between the two devices. The common shared secret is called a *link key.* This link key is established in a special communications session called pairing. All paired devices (devices that have had a previous connection to establish security procedures) share a common link key. There are two types of link keys defined in the [9]: *unit keys* and *combination keys.*

A device using a unit key uses the same secret for *all* of its connections. Unit keys are appropriate for devices with limited memory or a limited user interface. During the pairing procedure the unit key is transferred (encrypted) to the other unit. Note that only one of the two paired units is allowed to use a unit key.

Combination keys are link keys that are unique to a particular *pair* of devices. The combination key is only used to protect the communication between these two devices.

Clearly a device that uses a unit key is not as secure as a device that uses a combination key. Since the unit key is common to all devices with which the device has been paired, all such devices have knowledge of the unit key.

[1] 802.11 also supports "Open System Authentication" which is essentially no authentication, in other words, everyone who requests an authentication is authenticated. In our view, this is not a security flaw. The implementation choice to not authenticate is a valid one in some situations, such as where security requirements are limited or authentication is provided at a higher network level. A similar choice is allowed in the Bluetooth wireless communications.

Consequently they are able to eavesdrop on any traffic based on this key. In addition, they could, in theory, be modified to impersonate other devices using the key. Thus, when using a unit key there is no protection against attacks from other devices with which the device has been paired. As a result, the Bluetooth SIG discourages the use of unit keys in secure applications.

Authentication is performed with a challenge response scheme utilizing the E1 algorithm. E1 is a modification of the block cipher SAFER+. The scheme operates as follows: The verifier issues a 128 bit long challenge. The claimant then applies E1 using the challenge, its 48-bit Bluetooth address, and the current link key. He then returns the 32 most significant bits of the128 bit result[2]. The verifier confirms the response, in which case the authentication has succeeded. In this case, the roles are switched and the same procedure is applied again, thereby accomplishing mutual authentication.

The Bluetooth challenge response algorithm differs from that used in 802.11b in very important ways. In 802.11b the challenge and response form a plaintext/ciphertext pair. This fact, combined with the simplicity of the encryption method (XOR), allow an intruder to easily determine the authentication key string by listening to one authentication procedure. In contrast, the Bluetooth authentication method never transmits the complete challenge response pair. In addition, the E1 algorithm is not easily invertible. Thus even if an attacker has recorded an authentication challenge response session, he cannot (directly) use this data to compute the authentication key.

5.0 Data Eavesdropping, 802.11

The Bluetooth standard does not use RC4 but rather the stream cipher E0, which is specifically designed to run over a Bluetooth wireless packet network. A unique encryption key is generated for each session, from which per-packet keys are derived, in a manner that avoids the problem in 802.11b caused by frequent reuse of per-packet keys.

Direct attacks on the E0 cipher are known but are of significant complexity. Jakobsson and Wetzel present two such attacks [6] the first is of 2^{100} complexity, the second is a “birthday-type attack” of 2^{66}overall complexity. Fluher and Lukas [7] present an attack using observed keystream and the public knowledge of the encryption mechanism used in E0 to compute the encryption key. Their method requires from $O(2^{73})$ to $O(2^{84})$, depending on how much cleartext is available for the algorithm. They contend that the upper limit of E0 is actually about 80 bits and question the extension of the E0 key size to 128 bits as suggested in the Bluetooth specification [8]. As discussed by Jakobsson and Wetzel [6], attacks with a high order of complexity are not of practical value, but may point the way

[2] The remaining bits are called the Authentication Ciphering Offset (ACO) and are used to derive the ciphering key for data encryption.

to a more efficient attack. As yet, a more efficient direct attack on E0 has not been reported.

Like RC4, E0 required a ciphering key. The ciphering key is computed as a hash of a random number, the link key and a byproduct of the authentication procedure the Authentication Ciphering Offset (ACO).

While the link key is also used to generate a ciphering key used for data encryption, it is not used for data encryption itself. This is a significant advantage over 802.11b in which the same key is used for authentication and encryption.

In summary the known attacks on the E0 cipher used in Bluetooth are far more computationally complex then corresponding attacks on RC4 used in 802.11b. As yet, no practical direct attack has been reported. Also, unlike 802.11b, different keys are used for authentication and encryption. Accordingly practical studies on Bluetooth security have been focused on methods to guess or steal the key (or at least a portion of it). The most logical time to attempt this is during the pairing procedure.

6.0 Bluetooth Pairing

As discussed in Section 4.0 pairing is the procedure where a relationship (link key) is established between two previously unknown devices. The link key is *derived* when the devices are initially paired (i.e. the link key does not exist before the pairing procedure). Pairing is facilitated with yet another key, the *initialization* key. This key is computed by a pair of devices using the Bluetooth addresses of each device, a random number, and a shared secret (PIN). Since it is only used in the initial pairing, the initialization key is only used once.

The initial pairing is the most profitable area of attack on a Bluetooth device. If the attacker can guess or steal the PIN during the initial pairing, then he can perform a much more efficient search to derive the link key. This search is further simplified if the communications occurring while the devices are paired is recorded [6]. For this reason the Bluetooth SIG strongly encourages the use of long, random PINs and suggests that pairing be performed only in a private place. Assuming that both devices have a man-machine interface (such as a keypad) it is also suggested that the PIN be manually entered into both devices or in any case communicated out-of-band (not transmitted over the Bluetooth wireless link). Thus, long PINs provide improved security since the PIN cannot be received over-the-air. To steal the PIN an attacker must guess or record it by some other means such as direct observation of the user, a more difficult procedure if the PIN is long and the pairing is performed in private.

7.0 Final Comments

The known attacks on 802.11b security have been discussed and found not to apply to Bluetooth wireless technology. In particular

a) 802.11b authentication is highly susceptible to impersonation by recording only one authentication procedure. This is facilitated because a plaintext/ciphertext pair is transmitted. Bluetooth communications do not share this limitation.
b) 802.11b encryption is not very secure. The RC4 implementation used in 802.11b has several well-known direct attacks. Currently known direct attacks on the Bluetooth encryption are computationally complex and of little practical value.

From the preceding discussion it is clear that the weakest link in the Bluetooth security architecture is the initial pairing especially if a weak PIN is used. Accordingly the Bluetooth SIG strongly encourages pairing in a private place and the use of robust PINs. In addition, simple devices that use unit keys should not be relied upon to communicate highly secure data.

As a communication standard, Bluetooth security focuses on the link level. It provides both entity authentication and link privacy. Since these functions are focused at the lower network layers, message authentication and secure end-to-end links are not provided. However, many applications, such as e-mail and browser transactions require end-to-end security. As with other communication standards, this function is expected to be provided at higher network layers by specific application providers. Accordingly, the Bluetooth SIG encourages the reuse of existing transport, session and application layer security.

Regarding the security limitations that have been reported for 802.11b; the WLAN community is currently examining these issues. We expect them to be resolved with subsequent revisions of the standard. In addition several 802.11b vendors have added proprietary authentication and encryption procedures at higher network layers.

Acknowledgements

The authors wish to thank the members of the Bluetooth Security Experts Group and also Brent Miller and Peter Lee from IBM for their helpful comments.

References

[1] W. A. Arbaugh, "Wireless Research", available from, http://www.cs.umd.edu/~waa/wireless.html

[2] J. Walker, "Unsafe at any key size; An analysis of the WEP encapsulation", available from http://grouper.ieee.org/groups/802/11/Documents/DocumentHolder/0-362.zip

[3] N. Borisov, I. Goldberg, and D. Wagner, "Intercepting Mobile Communications: The Insecurity of 802.11." available from http://www.issac.sc.berkley.edu/issac/wep-faq.html.

[4] S. Fluhrer, I. Mantin, A. Shamir, Weaknesses in the Key Scheduling Algorithm of RC4, in the Workshop Record of SAC 2001

[5] W. Arbaugh, N. Shankar, Y.C.J. Wan, "Your 802.11 Wireless Network has No Clothes" available from http://www.cs.umd.edu/~waa/wireless.pdf

[6] M. Jakobsson and S. Wetzel, "Security Weaknesses in Bluetooth" available from http://www.rsasecurity.com/rsalabs/staff/bios/mjakobsson/bluetooth/bluetooth.pdf

[7] S. Fluhrer and S. Lucks, "Analysis of the E0 Encryption System" available from S. Lucks' web site at http://th.informatik.uni-mannheim.de/People/Lucks/papers/e0.ps.gz , a gnu-zipped Postscript file.

[8] Bluetooth SIG, *Specification of the Bluetooth system, Profiles"* ,, Version 1.1, 1 February 22, 2001, available at http://www.bluetooth.com/.

[9] Bluetooth SIG, *Specification of the Bluetooth system, Core "*, Version 1.1, 1 February 22, 2001, available at http://www.bluetooth.com/.

[10] B. Miller, "IEEE 802.11 and Bluetooth wireless technology" available from http://www-106.ibm.com/developerworks/wireless/library/wi-phone/