Secure authorization, access control and data integrity in Bluetooth

L. Nguyen
University of Wollongong

R. Safavi-Naini
University of Wollongong, rei@uow.edu.au

Willy Susilo
University of Wollongong, wsusilo@uow.edu.au

Tadeusz A. Wysocki
University of Wollongong, wysocki@uow.edu.au

The Bluetooth standard has a provision for mutual authentication of connecting devices but not their actual users and allows access control during connection setup only. We propose a user authorization and pairing (UAP) application, that has the ability to perform authentication and authorization of users using role based model. The pairing procedure, which exchanges link key between devices, is also performed as a part of the user authorization process. The integrity of the message is guaranteed by using message authentication codes. We also extend an attack on a short PIN during the pairing procedure for devices compliant with the Bluetooth specification version 1.1.

SECURE AUTHORIZATION, ACCESS CONTROL AND DATA INTEGRITY IN BLUETOOTH

Lan Nguyen, Rei Safavi-Naini, Willy Susilo and Tadeusz Wysocki

Centre for Computer Security Research
School of Information Technology and Computer Science
University of Wollongong
Wollongong 2522, AUSTRALIA
`{ldn01, rei, wsusilo, wysocki}@uow.edu.au`

ABSTRACT

Bluetooth standard has a provision for mutual authentication of connecting devices but not their actual users and allows access control during connection setup only. In this paper, we propose a User Authorization and Pairing (UAP) application, that has the ability to perform authentication and authorization of users using role based model. Pairing procedure, which exchanges link key between devices, is also performed as a part of the user authorization process. Integrity of the message are guaranteed by using message authentication codes. We also extend an attack on a short PIN during pairing procedure for devices compliant with the Bluetooth specification version 1.1.

1. INTRODUCTION

Convergence of computing and communications has been promised for many years. One of the most important component technologies is wireless communications which enables the computing devices, such as notebook computers, personal digital assistants (PDA), "smart" phones, two-way pagers, digital cameras, etc., to communicate or synchronize with each other wirelessly. An important technology towards Personal Area Network (PAN) is Bluetooth. Bluetooth protocol enables devices to communicate from a short distance without wires and has recently been proposed as the standard for short distance wireless communications of mobile devices [2].

In addition to being a cable replacement, Bluetooth also provides some other features, such as wireless ad-hoc connection between two mobile devices and access point to the Internet connection [2]. To ensure usage protection and data confidentiality, Bluetooth has to provide security measures both at the application layer and the link layer [3], which are appropriate for peer environments.

Bluetooth provides device authentication, encryption and device authorization. Device authentication is a process performed for devices to verify 'who' is at the other end of the link [16]. Security of these procedures depends on a shared link key between devices. The link key is established by a procedure called "pairing", in which the same PIN is entered in the devices as the initial shared secret [3, 16]. Bluetooth encryption uses stream cipher to protect the confidentiality of the link [3]. Device authorization is the process of determining whether a device is allowed to have access to a particular service [16]. Bluetooth uses transmission scheme that provides a level of security in itself. Instead of transmitting over one frequency within the 2.4 GHz band, Bluetooth radios use a frequency-hopping spread spectrum (FHSS) technique, allowing only synchronized receivers to access the transmitted data [12].

Bluetooth security has been studied by several authors, for example [7, 16, 17]. It is realized that Bluetooth security has several security drawbacks. The first important thing is the way its authentication works. Authentication mechanism in Bluetooth is used to authenticate a device and not the user [16]. The local device cannot determine who is using the remote device. It means that if someone steals a PDA which has been authenticated with a laptop earlier, then he can use this PDA to access services in the laptop, for example transferring or deleting files. Secondly, Bluetooth does not define authorization separately for each service [16]. If authorization is to be applied in the Bluetooth architecture, changes in the security manager and the registration processes would be necessary.

These two drawbacks have been imposed as open problems in [16]. In this paper, we propose a solution to this problem by proposing a secure authentication protocol and implementing a role based access control for granting authorization.

The final problem is concerning the data integrity. Bluetooth implicitly provides data integrity via encryption or CRC check sum. However, it is known that encryption does not guarantee data integrity [10]. It is also known that CRC only protects data on noisy channels and it cannot be applied to a channel which is controlled by the adversaries [10]. It implies that data integrity is not guaranteed in Bluetooth.

In our proposed protocol implementation, we solve these problems in a User Authorization and Pairing (UAP) Application. The application can authenticate and authorize users, who use a remote device and wish to access to services at the local device. Pairing is performed as part of User Authorization procedure, which is more secure because human interaction is not the main component in this procedure. Moreover, it allows a combination of message authentication and encryption to protect the communication.

We also note that there is a problem concerning the "pairing" mechanism. To perform a pairing, each user has to enter the PIN code twice on the unpaired devices, which is impractical. The security of the communication relies on the PIN, so the PIN is required to be random, secure and long enough [17]. Jacobson and Wetzel presented an attack if the PIN is not sufficiently long [7]. The attack is applicable to Bluetooth Specification version 1.0. In this paper, we also present an extension of this attack so that it is compliant with Bluetooth Specification version 1.1. As our UAP generates sufficiently long and random common PIN, it resists against this attack.

The rest of this paper is organized as follows. In the next section, we briefly review the Bluetooth Security described in [3]. In section 3, we present the extension of the attack of [7] which is applicable to Bluetooth Specification version 1.1. Section 4 presents some solutions to the Bluetooth Security problems stated in [16]. In section 5, we present our implementation of User Authorization and Pairing (UAP) Application, which has the ability to ensure data integrity in Bluetooth message communication. Section 6 concludes the paper.

1.1. Notations

The ring of integers modulo a number n is denoted by Z_n, and its multiplicative group, which contains only the integers relatively prime to n, by Z_n^*.

2. BLUETOOTH SECURITY

There are four different entities that are used to maintain security at the link layer: a 48 bits public address, BD_ADDR, which is unique for each device, two private user keys: 128 bits for authentication and 8 to 128 bits for encryption, and a 128 bit random number (RAND) which is different for each new transaction. The authentication key is used to generate the encryption key, each time encryption is activated and a new encryption key is requested. Therefore, the authentication key is more static in its nature compared to the encryption key. To underline the importance of the authentication key to a specific Bluetooth link, it is referred to as the *link key* [3].

The link keys are either semi-permanent or temporary [3]. A semi-permanent key is stored in non-volatile memory, that can be used after the current session terminates. A temporary link key is only available during the current session, and it cannot be reused after the session terminates. We omit the details of the link keys because it is irrelevant to our discussion and refer the reader to [3] for more details.

The next important thing to Bluetooth security is a PIN code. A PIN code is a number which can be fixed or selected by the user. The length of a PIN code is between 1 to 16 octets. The PIN must be entered to both devices during the pairing process [17].

2.1. Device Authentication

The device authentication in Bluetooth uses a challenge-response scheme, in which a claimant's knowledge of a secret key is verified through a 2-move protocol [3]. The verification involves the claimant's Bluetooth address (denoted by BD_ADDR_C), a random input, which is the *challenge*, denoted by AU_RAND_S, together with a link key, LK, and an authentication code, denoted by E_1. The result of the computation is $SRES$, which is then sent to the verifier device to be compared with its own calculated $SRES$ using its own inputs. The result of the verification implies that a correct claimant/verifier pair share the same secret key LK. More precise interaction is illustrated as follows.

1. $\mathcal{V} \rightarrow \mathcal{C} : AU\_RAND_S$
2. $\mathcal{C}$ computes $SRES = E_1(LK, BD\_ADDR_C, AU\_RAND_S)$
3. $\mathcal{C} \rightarrow \mathcal{V} : SRES$
4. $\mathcal{V}$ verifies whether $SRES \stackrel{?}{=} E_1(LK, BD\_ADDR_C, AU\_RAND_S)$ holds.

In the above interaction, $\mathcal{V}$ denotes the verifier and $\mathcal{C}$ denotes the claimant.
The authentication function E_1 is a computationally secure authentication code which uses an encryption function called SAFER+ [3]. The algorithm is an enhanced version of the existing 64-bit block cipher SAFER. The block cipher maps

$$\{0,1\}^{128} \times \{0,1\}^{128} \mapsto \{0,1\}^{128}$$

and the function E_1 uses this block cipher with the following mapping

$$\{0,1\}^{128} \times \{0,1\}^{128} \times \{0,1\}^{48} \mapsto \{0,1\}^{32} \times \{0,1\}^{96}$$

The outputs of this function are $SRES$ which is used in the verification procedure described above and ACO which is used for encryption procedure.

2.2. Encryption

The Bluetooth specification describes the link encryption algorithm as a stream cipher, which uses Linear Feedback Shift Registers (LFSRs). The effective key length is selectable between 8 and 128 bits to accomodate the security regulation applied in different countries. The encryption key, K_C, is derived by algorithm E_3 [3] from the current link key LK, a 96-bit Ciphering OFfset number (COF) and a 128-bit random number. The value of COF is determined as follows.

$$COF = \begin{cases} BD\_ADDR_{master} \cup BD\_ADDR_{master} \\ \quad \text{if } LK \text{ is a temporary key} \\ ACO \\ \quad otherwise \end{cases}$$

BD_ADDR_{master} is the address of master device in a point-to-multipoint session.

We note that because K_C depends on LK, and LK only authenticates the device but not the user, then K_C also authenticates the device only.

3. AN ATTACK ON SHORT PIN IN BLUETOOTH SPECIFICATION VERSION 1.1

When two devices do not have a common link key, an initialization key must be created based on a PIN and a random number. Then, an authentication needs to be done. The calculation of the authentication response is based on the initialization key. After a successful authentication, the link is created.

As pointed out in [7], by eavesdropping the pairing procedure between two victim devices, for instance BT_1 and BT_2, an attacker can derive the link key if the PIN entered is not sufficiently long. In this section, we will extend the attack so that it works with Bluetooth Specification version 1.1.

According to Bluetooth Specification version 1.1, the pairing procedure between BT_1 and BT_2 is done as follows.

1. BT_1: PIN is entered and initialization key is generated.
2. $BT_1 \rightarrow BT_2$: A random number to generate initialization key, K_{init}.
3. BT_2: PIN is entered and initialization key K_{init} is generated.
4. $BT_2 \rightarrow BT_1$: Indication to accept the pairing.
5. $BT_1 \rightarrow BT_2$: The first random number to generate combination key.
6. $BT_2 \rightarrow BT_1$: The second random number to generate combination key.
7. BT_2: Generate combination key.
8. BT_1: Generate combination key.
9. $BT_1 \rightarrow BT_2$: A random challenge for BT_1 to authenticate BT_2.
10. $BT_2 \rightarrow BT_1$: Response to the challenge.
11. $BT_2 \rightarrow BT_1$: A random challenge for BT_2 to authenticate BT_1.
12. $BT_1 \rightarrow BT_2$: Response to the challenge.

In step 1, 2, 3 and 4, a PIN and a random number are exchanged, then a shared initialization key is generated in both devices. In 5 and 6, two random numbers that are encrypted by K_{init} are exchanged. A combination key is generated from those random numbers in 7 and 8. In 9 and 10, BT_1 authenticates BT_2. In 11 and 12, BT_2 authenticates BT_1.

Hijacking attack can be done by an adversary who eavesdrops the traffic. The adversary will first guess all PINs up to a certain length (which must not be too large). From each guessed PIN, he calculates the initialization key, then the random numbers in 5 and 6, then the combination key and then computes the response to the challenge. He verifies the correctness of his guess by comparing his generated response and the transmitted response in 10. After reaching the correct PIN, he has the correct link key. Having the link key, he can eavesdrop transmissions between the two devices or impersonate one device to communicate with the other.

A similar attack can be performed, when the adversary initiates communication with the victim device. The adversary sends a random number to generate the initialization key K_{init}. Next, the adversary sends a random challenge to authenticate the victim device (similar message sequence as in the procedure above with BT_1 acting as the adversary). After receiving response from the victim device, the attacker will test all PINs up to a certain length by the response. An example situation where the attack can be applied is when the victim device has a fixed PIN. The adversary wants to access the device, but he does not know the correct PIN.

4. AUTHORIZATION AND ACCESS CONTROL IN BLUETOOTH

As mentioned earlier, Bluetooth specification does not authenticate the user, but the device itself. It is more desirable to have an authentication which will identify whether the user is authorized to access the device rather than only authenticate the device itself. In this section, we propose a user authentication mechanism that can be used on top of the Bluetooth device authentication.

The second problem is how to have an access control in Bluetooth? Consider a situation where a Bluetooth enabled laptop, which can provide FTP, LAN access and Serial Emulation, is connected with several devices owned by different people. So far, there is no easy way to allow a subset of these services to be accessed by different people with Bluetooth enabled devices. Once a Bluetooth enabled device is connected to the laptop, the Service Discovery Protocol (SDP) can be launched to search the available services, and all of them will be presented to the user. It would be desirable to have a role based mechanism that can identify whether a user is allowed to access a particular service provided, and allow to vary the access privileges for different user.

4.1. User Authentication

We use a password-based mutual authentication and key agreement protocol. The local device can authenticate the remote user and vice versa using this mechanism. The authentication is based on the knowledge of a *password.* A key is exchanged in the protocol to protect the communication session between the remote user and the local device.

There are several previous works on password-based mutual authentication and key agreement, such as [1, 6, 8, 9, 18]. Based on the comparison in [8], AuthA [1] is the protocol with the lowest communication cost. SRP [18] and AMP [8] require the lowest computation cost while still maintaining a pretty low communication cost. In our implementation, we use SRP as computation cost would be a very important factor for low-cost and power-constrained Bluetooth devices.

In the following, we briefly review the description of SRP. The protocol works in Z_n^*, where n is a large safe prime number. Let g denotes a primitive root modulo n. All of the computations in this protocol are performed modulo n. The server $\mathcal{S}$ stores the clients' public key v in its database (where $v = g^x$ and x is the client's secret key, which is obtained from $x = H(s, P)$, where s is a random string used as the user's *salt* and P is the user's password and H denotes a one-way hash function) together with a *salt* s, which is a random string. Suppose a client $\mathcal{C}$ wants to authenticate to a server $\mathcal{S}$, then the following protocol will be performed.

1. Firstly, $\mathcal{S}$ looks up (s, v) and sends s to $\mathcal{C}$.
2. $\mathcal{C}$ computes $x = H(s, P)$ and $A = g^a$, where a is an Ephemeral private key and it is generated randomly, and sends A to $\mathcal{S}$.
3. $\mathcal{S}$ computes $B = v + g^b$, where b is an Ephemeral private key and it is generated randomly, and generate another random number u.
4. $\mathcal{S}$ sends (B, u) to $\mathcal{C}$.
5. $\mathcal{C}$ computes $S = (B - g^x)^{a+ux}$ and $K = H(S)$.
6. $\mathcal{S}$ computes $S = (Av^u)^b$ and $K = H(S)$.
7. $\mathcal{C}$ computes $M_1 = H(A, B, K)$ and sends it to $\mathcal{S}$.
8. $\mathcal{S}$ verifies whether $M_1 \overset{?}{=} H(A, B, K)$ holds to authenticate $\mathcal{C}$.
9. $\mathcal{S}$ computes $M_2 = H(A, M_1, K)$ and sends it to $\mathcal{C}$.
10. $\mathcal{C}$ verifies whether $M_2 \overset{?}{=} H(A, M_1, K)$ holds to authenticate $\mathcal{S}$.

At the end of the protocol, $\mathcal{S}$ and $\mathcal{C}$ share the session key K.

4.2. Access Control

The purpose of having an access control is to grant or deny a particular service to a Bluetooth enabled user after the user authenticates himself to the other device. At the moment, Bluetooth only allows access control at connection set-up [16] and it is not granted on a role-based. In this section, we propose two solutions to this problem.

Trivial Solution: Access Control List
Suppose there is a Bluetooth enabled laptop $\mathcal{S}$ which has several services $S_1, S_2, \cdots, S_n$. We assume that there are n users $\mathcal{C}_1, \mathcal{C}_2, \cdots, \mathcal{C}_n$ who are authorized to use the service provided by the laptop after being authenticated and authorized. Each user $\mathcal{C}_i$, $i \in \{1, \cdots, n\}$ can have different services that are authorized. A trivial solution would be equipping the laptop with a set of access control list that specifies what are the services that will be granted when a user $\mathcal{C}_i$ connects to the system. An example of access control list is as shown as in Table 1.

$\mathcal{C}_1$	S_1, S_3
$\mathcal{C}_2$	S_2, S_3, S_n
$\cdots$	$\cdots$
$\mathcal{C}_n$	$S_1, S_2, \cdots, S_n$

Table 1. An Example of Access Control List

In the above example, user $\mathcal{C}_1$ can only request services S_1 and/or S_3 after he is authenticated and authorized by the laptop. On the other hand, user $\mathcal{C}_n$ can request any one of the services provided by the laptop.

Using this method, the number of data stored (i.e. the number of "rows" in the above table) will be linear to the number of users and the number of services. Hence, if we denote the number of users by $\tilde{c}$ and the number of services by $\hat{s}$, then the storage required using this method is $O(\tilde{c}\hat{s})$.

Role-based Access Control

In this section, we employ a role-based access control instead of the access control list, with the intention of improving the storage required by the Bluetooth enabled laptop which provides the services. Role-based Access Control (RBAC) is a powerful approach [15] and can be used [14] to model a variety of security policies and to simulate traditional methods.

For our purpose, we define three basic element sets in an RBAC model, namely $\mathcal{U}$ is a set of legitimate users, $\mathcal{O}$ is a set of roles assigned to each user $\mathcal{U}_i \in \mathcal{U}$ and $\mathcal{S}$ is a set of services available. In Bluetooth enabled device, which provides some services $S_i \in \mathcal{S}, i = 1, 2, \cdots, n$, there are two RBAC tables which will be used to either grant/deny a request by an authenticated user $\mathcal{U}_i$ in the system. An example of RBAC table is shown in Table 2.

$\mathcal{O}_1$	U_1, U_3
$\mathcal{O}_2$	U_2, U_6
$\mathcal{O}_3$	U_4, U_5

S_1	$\mathcal{O}_1, \mathcal{O}_3$
S_2	$\mathcal{O}_2, \mathcal{O}_3$

Table 2. An Example of Role-based Access Control

In the above example, there are six users $\mathcal{U} = \{U_1, U_2, \cdots, U_6\}$, three roles $\mathcal{O} = \{\mathcal{O}_1, \mathcal{O}_2, \mathcal{O}_3\}$ and two services $\mathcal{S} = \{S_1, S_2\}$ provided by a Bluetooth enabled laptop. Role $\mathcal{O}_3$ can access both services, but $\mathcal{O}_1$ can only access S_1. This means that U_1 and U_3 can only access S_1 because their role is $\mathcal{O}_1$, but U_4 can access all the services because he has the role as $\mathcal{O}_3$.

If we denote the number of users by $\tilde{c}$, the number of roles by $\bar{o}$ and the number of services by $\hat{s}$, then the storage required using this method is $O(\tilde{c} + \bar{o}\hat{s})$. We note that the storage required to store the role-vs-user table is $\tilde{c}$, because at the worst case, each role will be assigned to each user. The storage required to store service-vs-role table is $\bar{o}\hat{s}$ because each service can be granted multiple roles. We also note that compared to access control list method this method reduces the storage requirement.

5. USER AND PAIRING AUTHORIZATION (UAP) APPLICATION

The goal of our UAP application is to provide the following capabilities.

- Secure User Authentication and Authorization.
- Secure Pairing as part of the UAP procedure.
- Message Authentication to provide Data Integrity.

We tested our UAP application using Ericsson Bluetooth Application and Training Toolkit with ROK1008 hardware and PCR2B Host Stack, and the program is written in Visual C++ 6.0.

UAP consists of two main parts: UAP Server and UAP Client. UAP server is a service that uses Serial Port profile in the device, which has other services requiring user authorization. This device is called *Server Device*. Serial Port Profile defines the requirements for establishing emulated serial port connections using RFCOMM between two Bluetooth devices [5]. RFCOMM is a multiplexing protocol, providing emulation of serial ports [13, 11]. Services uses RFCOMM interface are assigned Server Channel numbers in the range of $1, \cdots, 30$ [4]. UAP server uses one RFCOMM's Server Channel. This service never requires User Authorization. UAP client is installed in the device where the user is using to access services in the server device. This device is called *Client Device*.

User Authentication

When a user uses a Client Device to log-in to a Server Device, a mutual authentication procedure will be performed. As mentioned in the previous section, we use SRP for this purpose. We assume that the Server Device protect the password verifier securely. The user proves his identity by showing his knowledge of the password. The Server Device proves its identity to the user by showing its knowledge of the password verifier (obtained in the initialization phase, when the user is registered to the server device).

UAP Server maintains the following information in its database for user authentication:

- *Authentication data*: including user names, salts, verifiers, modulus and generators for SRP.
- *Authenticated users*: list of the previous authenticated users. It has two fields: user name and remote device Bluetooth address.

Access Control

UAP uses RBAC as described in the previous section. Each service S_i has an associated role in $\mathcal{O}$. A user assigned to this role can access the service provided for this role.

Message Authentication

We use the 20-byte MAC incorporated in each message on RFCOMM to provide message authentication. For this purpose, each message is appended by a 20-byte Most Significant Bit (MSB) of the encrypted message using the session key K derived during the authentication process using SRP. On the receiver/verifier side, the message is encrypted again using the same session key and it is verified whether the 20-byte MAC matches with the calculated MAC to guarantee the data integrity in the message.

6. CONCLUSIONS

We extended the short PIN attack during the pairing procedure from [7] so that it can be successful for devices compliant with the Bluetooth specification version 1.1. We demonstrated our design and implementation of User Authorization and Pairing (UAP) application, which provides secure authorization, access control of services based on identify of the user as well as data integrity checker using MAC to protect the communication.

7. REFERENCES

[1] M. Bellare and P. Rogaway. The autha protocol for password-based authenticated key exchange.

[2] Bluetooth SIG. The Official Bluetooth Wireless Info Site. *http://www.bluetooth.com/.*

[3] Bluetooth SIG. Bluetooth Security. *Bluetooth Specification Version 1.1*, pages 148 – 180, 2001.

[4] Bluetooth SIG. RFCOMM with TS 07.10. *Bluetooth Specification Version 1.1*, 2001.

[5] Bluetooth SIG. Serial Port Profile. *Bluetooth Specification Version 1.1*, 2001.

[6] D. P. Jablon. Strong password-only authenticated key exchange. *Computer Communication Review, ACM*, 26(5):5–26, 1996.

[7] M. Jacobson and S. Wetzel. Security Weaknesses in Bluetooth. *RSA Conference 2001*, pages 179 – 191, 2001.

[8] T. Kwon. Authentication and key agreement via memorable password, 2000.

[9] P. MacKenzie and R. Swaminathan. Secure network authentication with password identification. Submitted to IEEE P1363a, 1999.

[10] A. J. Menezes, P. C. van Oorschot, and S. A. Vanstone. *Handbook of Applied Cryptography*. CRC Press, Boca Raton, New York, 1997.

[11] B. A. Miller and C. Bisdikian. *Bluetooth Revealed.* Prentice Hall, 2002.

[12] MobileInfo. Bluetooth Technology. *http://www.mobileinfo.com/bluetooth/.*

[13] N. J. Muller. *Bluetooth Demystified.* McGraw-Hill Telecom, 2001.

[14] S. Osborn, R. Sandhu, and Q. Munawer. Configuring Role-based Access Control to Enforce Mandatory and Discretionary Access Control Policies. *ACM Transactions on Information and System Security, Volume 3, Number 2*, pages 85 – 106, 2000.

[15] R. Sandhu and P. Samarati. Access Control: Principles and Practice. *IEEE Communications, Volume 32, Number 9*, 1994.

[16] M. Träskbäck. Security of Bluetooth: An Overview of Bluetooth Security. *Technical Report, http://www.cs.hut.fi/Opinnot/Tik-86.174/Bluetooth_Security.pdf.*

[17] J. T. Vainio. Bluetooth Security. *Proceedings of Helsinky University of Technology, Telecommunications Software and Multimedia Laboratory, Seminar on Internetworking: Ad Hoc Networking, http://www.niksula.cs.hut.fi/ jiitv/bluesec.html*, 2000.

[18] T. Wu. The Secure Remote Password Protocol. *Proceedings of the 1998 Internet Society Network and Distributed System Security Symposium*, pages 97 – 111, 1998.