A Service-Level Security Protocol for Bluetooth Scatternets

Fathi Taibi[1] and Toufik Taibi[2]
[1]Faculty of Information Technology, University of Tun Abdul Razak, Malaysia
[2]College of Information Technology, United Arab Emirates University, UAE

Abstract: *Bluetooth is the term used to describe the protocol of short-range, and frequency hopping radio link between devices. Bluetooth enables users to connect wirelessly to a wide range of computing and communication devices easily and it provides opportunities for rapid ad hoc connections. In this type of networks, security is one of the main issues. Bluetooth link-level security provides authentication of the communicating participants and exchange of encrypted data at the link-level. The existing Bluetooth security mechanism has limitations regarding strength, flexibility and efficiency. In this paper, a critical analysis on Bluetooth link-level security mechanism is provided, and a new Bluetooth service-level security protocol is proposed to overcome the limitations of the existing security mechanism.*

Keywords: *Bluetooth, scatternet, service-level, security protocol.*

Received September 27, 2005; accepted March 2, 2006

1. Introduction

Bluetooth [1, 2, 4, 5] is an emerging low cost and low power radio technology. It is the term used to describe the protocol of a short-range (10 to 100 meters) frequency-hopping radio link between Bluetooth-enabled devices. A Bluetooth-enabled device is an electronic device equipped with a Bluetooth radio module. The members of the Bluetooth Special Interest Group (SIG) released early revisions of the specification and version 1.0 was published in 1999. Since then, it has been progressively updated, and the latest version is 2.0.

Bluetooth enables users to connect to a wide range of computing and telecommunication devices easily and simply, without the need to buy, carry, or connect cables. It delivers opportunities for rapid ad hoc connections, and the possibility of automatic, unconscious, connections between devices. The communication over Bluetooth is divided into several communication layers. Bluetooth provides a point-to-point connection, or a point to multipoint connection. The channel is shared among several Bluetooth units. Two or more units sharing the same channel form a piconet. One Bluetooth unit acts as the master of the piconet, whereas the other units act as slaves [1]. Two or more overlapping piconets form a scatternet and inter-piconets communication are facilitated via sharing slaves. Figure 1 shows an example of a Bluetooth scatternet.

Bluetooth technology adoption is wide spreading throughout the computer and telecommunications industry. As Bluetooth wireless technology is incorporated in more personal mobile devices, it enables new uses for those devices. One of such uses is to use a mobile device as a method of payment for goods and services, or to use Bluetooth links to create a home networks to connect (wirelessly) electronic devices such as hand-phones, laptops, and wireless access points.

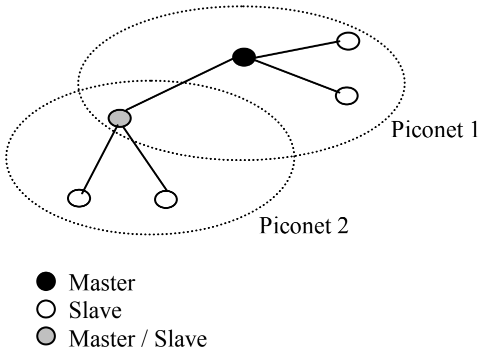


Figure 1. A Bluetooth scatternet.

For all the applications of Bluetooth technology, there is always a need for security to protect users' data. As with Bluetooth technology the data is transmitted over the air, this makes an increase need for security because anyone with proper equipments can sniff the data transmitted over the air, and this makes Bluetooth security one of the main issues to consider when adopting this technology.

Bluetooth security has become increasingly important since Bluetooth became a standard technology in wireless personal communication [9], especially for security-sensitive applications. Bluetooth link-level security provides authentication of the communicating participants and exchange of encrypted data at the link-level, i. e., before a device receives a channel establishment request from a remote device [7]. The overall security mechanism has limitations that have to be overcome to make Bluetooth connections more secure. In this paper, a critical analysis of the existing Bluetooth security mechanism is presented first, followed by the adoption of a Bluetooth service-level security architecture and the definition of a new security procedures and parameters that will help to overcome the limitations of the existing security mechanism. Finally, a protocol is proposed to specify how Bluetooth security architecture layers interact in order to perform the security check.

2. A Critical Analysis on Bluetooth Link-Level Security Mechanism

Bluetooth link-level security provides security procedures (authentication and encryption) at the link-level. Bluetooth authentication is a challenge response scheme where a claimant's unit knowledge of a secret key is verified. Authentication is performed using a link-key, a 128-bit long private key shared between two or more parties. This key is used for all security transactions between them. Bluetooth encryption system is a symmetric stream cipher, it uses an encryption key (up to 128-bit length) that is generated based on the link-key; a new encryption key is generated each time encryption is activated.

The study done on Bluetooth link-level security has shown that its authentication and encryption procedures have some strength as follows:

1. The authentication procedure is based on a shared link-key. If the link key is not known to anyone of the devices then authentication fails. If the link-key used is a combination key (combining two private keys generated by both devices) then it makes it very difficult for a third party (unauthorized device) to access it and consequently makes the authentication procedure more secure.
2. Bluetooth security mechanism allows mutual authentication i. e., provides a claimant device with the ability to authenticate back a verifier device after a successful authentication check process, which strengthens the security mechanism.
3. The use of a master key as a temporary link key if the master device needs to send the same data to several slaves is efficient because using this temporary link key all the piconet's slaves can generate the encryption key and use it to decrypt the data received from the master device.
4. Bluetooth encryption procedure uses a binary key stream (K_{cipher}) that is used to encrypt the data to be sent. A new K_{cipher} is generated for every payload (packet), which makes it very difficult for a third party to get it. Bluetooth encryption is also efficient because the same key is used to encrypt and decrypt the data sent (symmetric).

However, Bluetooth link-level security has limitations regarding strength, flexibility, and efficiency as follows:

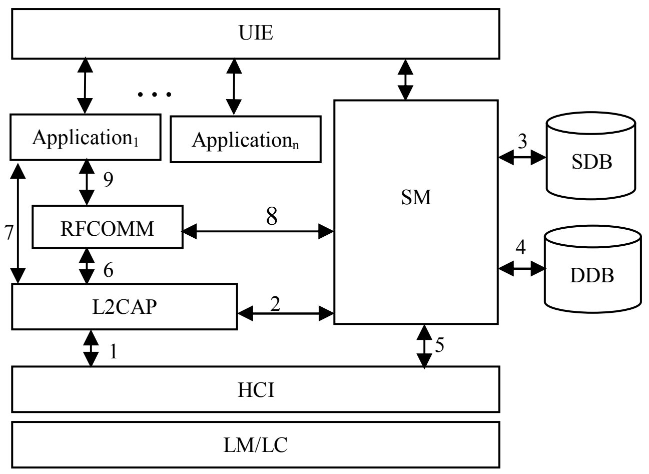
1. The initialisation key that is used to generate the link key is mainly generated using a PIN (Personal Identification Number) code that is a four decimal digits code. A third party can have access to the initialisation key through PIN code guessing as the number of possible PIN combinations is limited to 10000 possibilities. In case a third party accesses the PIN code, it will succeed in all the security procedures that follow and will be able to have access to the sensitive data exchanged. This problem can be overcome by choosing the PIN length to be between 1 and 16 bytes; in this case, the units have to exchange PIN codes through means supported by software at the application layer [9].
2. Bluetooth encryption key is a variable length key (8-128 bit), an attacker's task is facilitated in case the encryption key length is chosen to be less than 128-bit because the number of possible values of that key decreases. This vulnerability can be overcome by adopting encryption keys that are at least 128-bit length.
3. There is no consideration about what service is requested by the remote device, and what are the security procedures needed to access it. This makes the overall security mechanism less efficient because some services may need strong security procedures while some other services may not need any security procedure at all.
4. Bluetooth link-level security check process does not require any input from the user (or any application acting on behalf of him) except for PIN entering. This makes it less flexible and less robust because the robustness of any security procedure requires some inputs from a higher authority, e.g. the owner of the device, which are used in the security check. In addition, the security mechanism is more flexible when the device user is allowed to set or change the security parameters.
5. Bluetooth link-level security does not provide means for a device to set remote devices trust-level within the local device. This is needed in order to create or remove a trusted relationship with a remove device that is needed to make the future connections between them easier.
6. There is no support for authorization, which is needed to enforce security when used along with authentication and encryption. Authorization procedure may need some form of user-interaction to add strength to the security procedure.

3. Bluetooth Service-Level Security Architecture

Bluetooth link-level security limitations that have been highlighted in the previous section need to be addressed in order to strengthen the security mechanism. The solution is to adopt a service-level security mechanism that is intended for applications that require strong security implementations, and to complement the link-level security mechanism by improving its strength, flexibility, and efficiency. To achieve this, new security parameters will be defined and used in a new authorization procedure, and a protocol will be proposed to define how the security architecture layers interact when performing the security check.

In the adopted Bluetooth security architecture [7], the security is maintained by a Security Manager (SM) entity that is responsible of granting (or denying) access to services. Granting access is based on the requested service security-level, and the remote device trust-level within the local device. A service security-level parameter determines the need for authorization, authentication, and encryption to access it. A remote device may be regarded as trusted or untrusted depending on previous connections.

The SM maintains databases with information about services available in the local device, and information about remote devices collected through previous connections. This information is used in the access check done by the SM when receiving a channel establishment request. Figure 2 shows the Bluetooth service-level security architecture.



UIE: User Interface Entity.
SM: Security Manager.
SDB: Services Database.
DDB: Devices Database.
RFCOMM: Radio Frequency Communication Protocol.
L2CAP: Logical Link Control and Adaptation Protocol.
HCI: Host Controller Interface.
LM/LC: Link Manager / Link Controller.

Figure 2. Bluetooth service-level security architecture and the access request information flow.

The SM maintains information about the available services in the local device in Services Database (SDB) [9]. This information is collected through querying a User Interface Entity (UIE). The key attribute in SDB is the unique Protocol Service Multiplexer (PSM) value that identifies any service available in a Bluetooth device. In addition, the requested service security-level parameter determines the needs for authorization, authentication, and encryption to access it.

The SM also maintains information about other known devices in Devices Database (DDB); this information is collected during previous connections. A Bluetooth Device Address (BD-ADDR) attribute is a unique 48-bit IEEE address identifying every Bluetooth device. In DDB, BD-ADDR attribute represents the Bluetooth device address of a remote device. A trust level attribute is used to state whether a given remote device is trusted within the local device or not. In order to create a trusted relationship, the trust-level attribute can be set to 1 by UIE through querying the SM. In addition, DDB has a 'failure_nbr' attribute that represents the number of connection attempt failures to the local device performed by the remote device identified by its BD-ADDR. Finally, DDB has a 'max_allowed' attribute that represents the maximum number of connection attempt failures allowed to the remote device identified by its BD-ADDR. Both, failure_nbr and max_allowed attributes are used in the authorization check process.

3.1. Access Request Information Flow

The access check procedure is initiated when L2CAP receives a channel establishment request from a remote device. Based on that, all the entities forming the security architecture start interacting in a request-response scheme to perform the security check.

The access request as shown in Figure 2 consists of the following steps:

1. L2CAP receives a channel establishment request to a specific application from HCI.
2. L2CAP requests access check from the SM.
3. SM looks up in SDB to determine the requested application (service) security-level.
4. SM looks up in DDB to determine the trust-level of the remote device.
5. Depending on the security information obtained from the two databases, the security manager starts an authorization procedure and/or requests authentication (and/or encryption) from the HCI.
6. If access is granted, L2CAP continues to set up the connection by requesting RFCOMM (if the later protocol is needed in the connection establishment).
7. If RFCOMM is not needed, L2CAP may continue establishing the connection with the requested application.
8. If requested by L2CAP, RFCOMM may request access check from the SM. The SM does not apply a new access check procedure but the result of the

previous access check requested previously by L2CAP is verified a second time.

9. In case of receiving a positive response from the SM, RFCOMM will continue establishing the connection with the requested application.

3.2. Access Check Procedure

After determining what are the security procedures that must be applied to grant access, i. e., establishing a channel with the remote device, the SM of the local device starts an access check procedure that takes into consideration the requested service security-level and the remote device trust-level within the local device. Figure 3 shows the flowchart of the access check procedure.

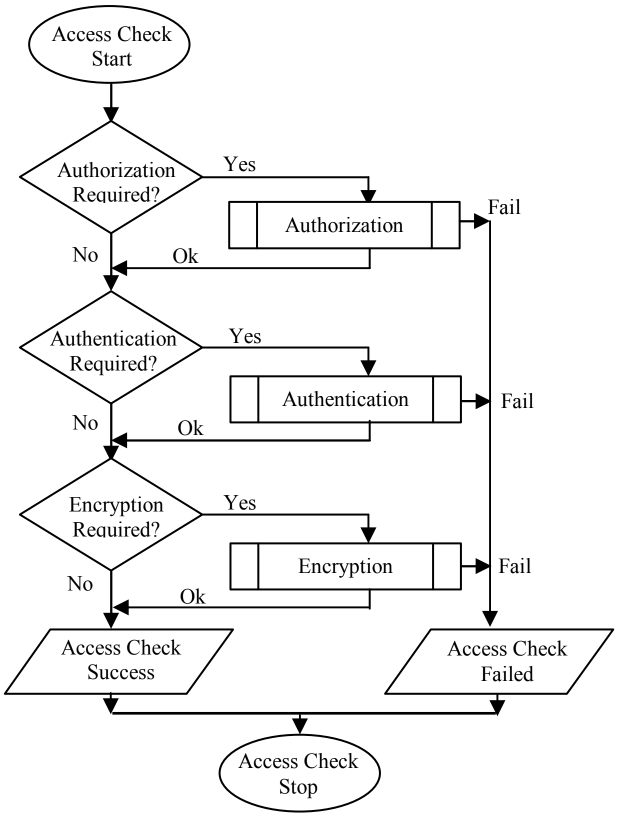


Figure 3. Access check flowchart.

If the remote device is trusted, authorization is not required otherwise, it is applied. The proposed authorization procedure consists of comparing the number of previous connection failure attempted by the remote device (failure_nbr) with the maximum number of failures allowed (max_allowed), if it is greater the authorization fails, otherwise it succeeds. Requesting directly the UIE to grant authorization is also possible. Before approving authorization failure, the SM requests the UIE to clear 'failure_nbr' or to change 'max_allowed'. If the request is approved then authorization is granted. The SM can also requests the UIE if a trusted relationship (i. e., marking the remote device as trusted in DDB) can be created. Figure 4 shows the flowchart of the authorization procedure.

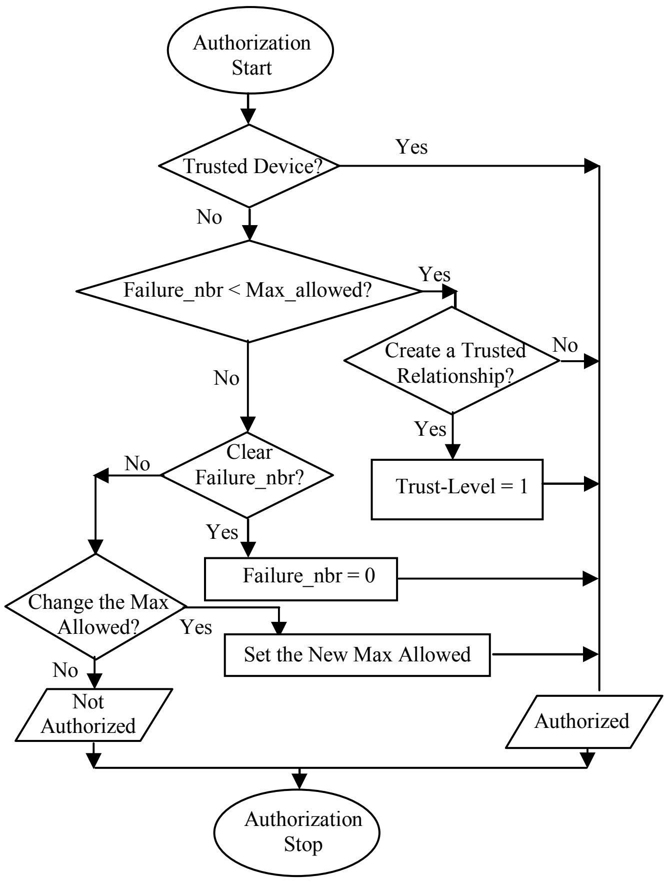


Figure 4. Authorization flowchart.

If a link-key has been created already, authentication can be applied. Otherwise, devices must go through a pairing procedure that consists of creating an initialization key based on the PIN code of the verifier device. The SM requests the HCI for authentication, and then HCI requests the LM to authenticate the remote device. The result of authentication is passed back to the SM via HCI. The SM may also request HCI for encryption, in that case, HCI requests the LM to use encryption for the current communication. The entire interactions between the security architecture entities are defined in section four.

The proposed service-level security mechanism improves the flexibility and the efficiency of the security check process by considering services security-level and devices trust-level parameters because the security procedures that should be applied are relative to the requested service and the requesting device. The proposed authorization procedure improves the strength of the security by allowing remote devices to be marked as 'untrusted' based on their number of access check failures. Remote devices can even be marked as 'barred' where they are not allowed to go through the access check procedure case their failure_nbr is high. The authorization procedure maintains also the flexibility of the security mechanism by allowing remote devices to be marked as 'trusted' so that the authorization procedure is not needed in the incoming connections between the local device and the one marked as 'trusted'. The authorization procedure maintains also the efficiency of the security mechanism by giving the UIE the possibility to be involved in the security check process by setting (or changing) max_allowed, failure_nbr, and devices trust-level.

4. An Interaction Protocol for the Security Architecture Entities

After defining the new security parameters and the new authorization procedure, a protocol need to be proposed to describe how the security architecture entities interact to perform the security check. The proposed protocol uses a request-response scheme to define how the security architecture entities interact in order to perform the security check. All transactions are presented as function calls with parameters and return values in some cases.

The interaction between RFCOMM and the other entities will not be discussed because not all applications make use of this protocol to establish a connection with a remote application that is located in a remote device. In addition, LM and the LC are addressed as Lower Protocols (LP) because a service-level security protocol is presented where the focus is more on how layers above HCI interact in order to perform a security check. Tables 1, 2, and 3 show all the functions used to define the interaction protocol between the security architecture layers during the security check process.

Table 1. Interactions between HCI and SM.

Function	Caller	Destination	Parameters and Returned Value
SM-Authentication-Req Used to request authentication.	SM	HCI	- BD-ADDR of the remote device and the link key. - No return.
HCI-Authentication-Rsp Used as a response to SM-authentication-req.	HCI	SM	- BD-ADDR of the remote device. - Returns succeeded / failed.
SM-Encryption-Req Used to request encryption.	SM	HCI	- BD-ADDR of the remote device and the link key. - No return.
HCI-Encryption-Rsp Used as a response to SM-encryption-req.	HCI	SM	- BD-ADDR of the remote device. - Returns succeeded / failed.

When the HCI receives a channel establishment request it sends it to L2CAP by calling the 'HCI-channel-Creation-req' function. Then, L2CAP requests the SM to start the access request check procedure by calling the 'L2CAP-access-check-req' function. The SM starts looking for the security parameters associated with the remote device (identified by its BD-ADDR) in DDB and SDB. Based on the request service security-level, the SM requests the HCI to authenticate the remote device and/or to use encryption by calling the functions 'SM-authentication-req' and 'SM-encryption-req'. After that, the HCI calls the functions 'HCI-authentication-req' and 'HCI-encryption-req' to ask LP to run the authentication and encryption procedures.

Table 2. Interactions between SM, L2CAP, and UIE.

Function	Caller	Destination	Parameters and Returned Value
L2CAP-Access-Check-Req Used to request applying security access check procedure.	L2CAP	SM	- BD-ADDR of the remote device and the PSM value of the requested service. - No return.
SM-Access-Check-Rsp Used as a response to L2CAP-access-check-req.	SM	L2CAP	- BD-ADDR of the remote device. - Returns succeeded / failed.
SM-Enter-Pin-Req Used to request PIN entering.	SM	UIE	- BD-ADDR of the remote device. - No return.
UIE-Enter-Pin-Rsp Used as a response to SM-enter-pin-req.	UIE	SM	- BD-ADDR of the remote device. - Returns the PIN code.
SM-Device-Trust-Level-Req Used to request a device trust-level.	SM	UIE	- BD-ADDR of the remote device. - No return.
UIE-Device-Trust-Level-Rsp Used as a response to SM-device-trust-level-req.	UIE	SM	- BD-ADDR of the remote device. - Returns the trust-level.
SM-Service-Security-Level-Req Used to request a service security-level.	SM	UIE	- SM value of the requested service. - No return.
UIE-Service-Security-Level-Rsp Used as a response to SM-service-security-level-req.	UIE	SM	- PSM value of the requested service. - Returns the service security-level.
SM-Max-Failure-Number-Req Used to request the max number of failures allowed entering.	SM	UIE	- BD-ADDR of the remote device. - No return.
UIE-Max-Failure-Number-Rsp Used as a response to SM-max-failure-number-req.	UIE	SM	- BD-ADDR of the remote device. - Returns the max_allowed.
SM-Clear Failure-Number-Req Used to request clearing the number of failures of a remote.	SM	UIE	- BD-ADDR of the remote device. - No return.
UIE-Clear Failure-Number-Rsp Used as a response to SM-clear-failure-number-req.	UIE	SM	- BD-ADDR of the remote device. - Returns succeeded / failed.
SM-Create-Trust-Req Used to request the creation of a trusted relationship.	SM	UIE	- BD-ADDR of the remote device. - No return.
UIE-Create-Trust-Rsp Used as a response to SM-create-trust-req.	UIE	SM	- BD-ADDR of the remote device. - Returns the new trust-level.

Table 3. Interactions between HCI, LP, and L2CAP.

Function	Caller	Destination	Parameters and Returned Value
LP-Connection-Req Used to request a channel creation.	LP	HCI	- BD-ADDR of the remote device and the PSM value of the requested service. - No return.
HCI-Connection-Rsp Used as a response to LP-connection req.	HCI	LP	- BD-ADDR of the remote device. - Returns succeeded/ failed.
HCI-Authentication-Req Used to request authentication.	HCI	LP	- BD-ADDR of the remote device and the link key. - No return.
LP-Authentication-Rsp Used as a response to HCI-authentication-req.	LP	HCI	- BD-ADDR of the remote device. - Returns succeeded/ failed.
HCI-Encryption-Req Used to request encryption.	HCI	LP	- BD-ADDR of the remote device and the link key. - No return.
LP-Encryption-Rsp Used as a response to HCI-encryption-req.	LP	HCI	- BD-ADDR of the remote device. - Returns succeeded/ failed.
HCI-Channel-Creation-Req Used to request a channel creation.	HCI	L2CAP	- BD-ADDR of the remote device and the PSM value of the requested service. - No return.
L2CAP-Channel-Creation-Rsp Used as a response to HCI-channel-creation-req.	L2CAP	HCI	- BD-ADDR of the remote device. - Returns succeeded/ failed.

The results from the authentication and encryption procedures are returned back to HCI when LP calls the functions 'LP-authentication-rsp' and 'LP-encryption-rsp'. The same results are returned back to the SM when HCI calls the functions 'HCI-authentication-rsp' and 'HCI-encryption-rsp'. The SM returns those results to L2CAP by calling the 'SM-access-check-rsp' function. Based on the results obtained, L2CAP might continue setting up the connection with the corresponding application.

In case there is no link-key associated with the remote device, the SM requests the UIE to enter the PIN code to create an initialization key. This is done by calling the 'SM-enter-pin-req' function. The UIE replies by calling the 'UIE-enter-pin-rsp' function. The SM can also request the UIE to get (or change) the other security parameters by calling the functions 'SM-device-trust-level-req', 'SM-service-security-level-req', 'SM-max-failure-number-req', 'SM-clear-failure-number-req', and 'SM-create-trust-req'. The UIE responds to the SM requests by calling the functions 'UIE-device-trust-level-rsp', 'UIE-service-security-level-rsp', 'UIE-max-failure-number-rsp', 'UIE-clear-failure-number-rsp', and 'UIE-create-trust-rsp'.

After receiving a positive channel creation response from the remote device that has performed the security procedures, the local device can decide to perform the same security check procedures (or the authentication procedure only) to the remote device. This is called mutual security check (or mutual authentication), and it makes the connection created between the devices more secure.

The proposed security architecture entities interaction protocol strengthen the security mechanism as the outcome of the security check process depends on all the architecture layers needed to create the channel with the remote device. In addition, the request-response model used contributes to the simplicity of the proposed protocol and consequently facilitates its future implementation.

5. Conclusion

In this paper, a new service-level security protocol has been proposed to overcome the limitations of Bluetooth link-level security. A flexible security architecture has been adopted that is based on a SM entity. The SM is responsible for managing the security check process in any Bluetooth-enabled device. The SM is requested to perform the security check process when a channel establishment request is received from a remote device and the security procedures are performed based on the requested service security-level, the remote device trust-level and other parameters (failures_nbr and max_allowed) collected from the previous connections with the remote device.

The paper's contribution lies in analyzing Bluetooth link-level security by identifying it's strengths and weaknesses, defining the new security parameters that are used in the proposed authorization procedure, and proposing a new protocol describing the interactions between the entities involved in the security check process. These interactions (requests and responses) are presented as function calls. Finally, a simulation software has been developed to evaluate the performance of the proposed security protocol, and the simulation results obtained are discussed in [9].

References

[1] Bluetooth Special Interest Group (SIG), "Bluetooth Core Specification," version 2.0, available at: http://www.bluetooth.org/spec/, 2006.

[2] Brent A. M. and Chatschile B., *Bluetooth Revealed: The Insider's Guide to an Open Specification for Global Wireless Communications*, Prentice Hall, 2000.

[3] Ching L., Mehta A. K., and Siu K., "Performance of a New Bluetooth Scatternet Formation Protocol," *in Proceedings of the ACM Symposium on Mobile Ad Hoc Networking and Computing*, California, USA, 2001.

[4] Haartsen J., “Bluetooth Radio System,” *IEEE Personal Communications*, vol. 7, no. 1, pp. 28-36, 2000.

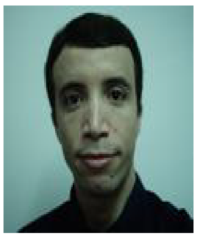
[5] Haartsen J., Naghshineh M., Inouye J., Joeressen O. J., and Allen W., “Bluetooth: Vision, Goals, and Architecture,” *ACM Mobile Computing and Communications Review*, vol. 2, no. 4, pp. 38-45, 1998.

[6] Lamm G., Falauto G., Estrada J., and Gadiyaram J., “Bluetooth Wireless Networks Security Features,” *in Proceedings of the 2001 IEEE Workshop on Information Assurance and Security*, United States Military Academy, West Point, NY, USA, 2001.

[7] Muller T., “Bluetooth Security Architecture”, White Paper, Version 1, 1999.

[8] Salonidis T., Bhagwat P., Tassiulas L., and LaMaire R., “Distributed Topology Construction of Bluetooth Personal Area Networks,” *in Proceedings of IEEE INFOCOM*, Anchorage, AK, USA, pp. 1577-1586, 2001.

[9] Taibi F., “A New Bluetooth Service-Level Security Protocol,” *Master Thesis*, Faculty of Computer Science and Information Technology, University of Malaya, Malaysia, 2003.



Fathi Taibi is a lecturer at the Faculty of Information Technology of the University of Tun Abdul Razak, Malaysia. He holds a BSc (Hon) degree and a Master degree, both in computer science. He has more than 6 years of teaching and research experience. His research interests include formal specification, security in mobile networks, and object-oriented methodologies.

Toufik Taibi is an assistant professor at the College of Information Technology at United Arab Emirates University, UAE. He holds a BSc, a MSc, and a PhD degrees, all in computer science. He has more than 12 years of teaching and research experience. His research interests include formal specification of design patterns, distributed object computing, and component-based software engineering.