Bluetooth WHITE PAPER	DATE	N.B.	DOCUMENT NO
	15 July 99		1.C.116/1.0

RESPONSIBLE	E-MAIL ADDRESS	STATUS
Thomas Muller	thomas.t.muller@nmp.nokia.com	

Bluetooth Security Architecture

Version 1.0

This White Paper describes a flexible security architecture for Bluetooth that allows different security levels for applications. While Bluetooth provides link-level authentication and encryption, enforcing at only this level prevents user-friendly access to more public-oriented usage models such as discovering services and exchanging business cards. This architecture uses the link-level security mechanisms of Bluetooth to enforce the service level security policy (security mode 2) of the Generic Access Profile.

Special Interest Group (SIG)

The following companies are represented in the Bluetooth Special Interest Group:
Ericsson Mobile Communications AB
IBM Corp.
Intel Corp.
Nokia Mobile Phones
Toshiba Corp.

Revision History

Revision	Date	Comments
0.0	1999-03-29	first draft, based on discussion at the SW face-to-face meeting in chandler, AZ
0.0.1	1999-03-30	Requirement on limited user intervention added 2 requirements in question added Start work on procedures (general behavior, Handling of RFCOMM)
0.0.2	1999-04-01	Incorporated feedback from Paul and Chatschik
0.0.3	1999-04-07	Feedback from Brian Redding
0.1	1999-04-09	Integrate decisions from the meeting 1999-04-08 Add interfaces of the security manager
0.2	1999-04-16	Modifications to the interfaces of the security manager: – Queries from L2CAP and other protocols harmonised – Only BD_ADDR used in query – Entity taking care of registration is implementation dependent. Registration moved to a separate section; interface to applications removed. – UI: set-up of trusted relationship included Security Policy for changed connection (section 2.1): wording changed to reflect that this includes client and server role. Section 3.1: – Pairing removed – registration can also be done by general management entity.

0.3	1999-04-27	– Remove parts for L2CAP connection hold after BB loss, because not supported by L2CAP any more: mainly changes in 3.5.2 and 3.5.3. – Flow chart changed according to phone meeting April 21st and included in document – Requirements for service security levels (requirement 3) corrected. Changes to distinguish between outgoing and incoming connections: – Default security level (in section 3.2.3) – Interface for registration: levels for both incoming and outgoing connections separately defined – Query to security manager: attribute for incoming/outgoing connection added Changes to make authentication mandatory in case authorisation is required: Statements in 3.2.1 and 3.2.3
0.5	1999-05-16	Security levels for registering multiplexing protocols added in section 3.6.5. Incorporate the changes agreed upon at the interoperability face-to-face meeting in Tampere: – Trust levels of devices might be set individually for services or groups of services. – Key management functions outside of Bluetooth mentioned – Trust flag replaced by more generic wording.
0.51	1999-05-26	Added statement on encryption in 2.1
0.8	1999-06-25	Introduction completely rewritten Requirements/Design objectives => what does the architecture provide Major editorial changes Removed chapter on consequences for Bluetooth specs Added section 4.6 Interface to HCI / Link Manager Added parameter ConnectionHandle in - 4.2 Interface to L2CAP - 4.3 Interface to other multiplexing protocols because it is needed in section 4.6 for HCI commands

0.86	1999-07-02	Incorporated changes from Chatschik and Jon • Abstraction: user ⇒ ESCE • Statement on application level security in Section 2.4 • Unknown device is also untrusted (Section 3.2.2) • Requirements for transition from security mode 2 to 3 added • Explanation for outgoing connections • Section 3.3.5.1 removed • Section 4.4: UI ⇒ ESCE and statements on calling directions
1.0	1999-07-13	Include PIN request to ESCE Terminology reference to GAP Replace initialization with bonding Editorial changes

Contributors

Paul Moran	3COM
Patric Lind	Ericsson
Patrik Olsson	Ericsson
Johannes Elg	Ericsson
Chatschik Bisdikian	IBM
Amal Shaheen	IBM
Jon Inouye	Intel
Robert Hunter	Intel
Brian Redding	Motorola
Stephane Bouet	Nokia
Thomas Müller (Owner)	Nokia
Martin Roter	Nokia

Disclaimer and copyright notice

Contents

1 Introduction 8

2 Functionality & Limitations 9

2.1 Security Levels 9
2.2 Flexibility and Usability 9
2.3 Implementation and Bluetooth-specific Matters 10
2.4 Risks and Limitations: 10

3 Security Architecture to meet the Requirements 12

3.1 Overview 12
3.2 Security Levels 13
3.2.1 Authorisation and Authentication 13
3.2.2 Device Trust Level 14
3.2.2.1 Authenticate trusted device 14
3.2.2.2 Set-up of the trusted relationship 14
3.2.2.3 Authenticate untrusted device 15
3.2.3 Security Level of Services 15
3.3 Connection Set-up Procedure 15
3.3.1 General Concept 15
3.3.2 Authentication on Baseband Link Set-up 16
3.3.3 Handling for Protocol Stack 17
3.3.4 Registration Procedures 18
3.3.5 External Key Management 19
3.4 Access Control Procedures 19
3.5 Connectionless L2CAP 22

4 Interfaces and Functions of the Security Manager 24

4.1 Databases 24
4.1.1 Service Database 24
4.1.2 Device Database 24
4.1.3 Temporary Storage 25
4.2 Interface to L2CAP 25
4.3 Interface to other multiplexing Protocols 26
4.4 Interface to ESCE (e.g., UI) 27
4.5 Registration Procedures 28
4.6 Interface to HCI / Link Manager 30
4.6.1.1 Authentication request 30
4.6.1.2 Encryption control 30
4.6.1.3 Name request to remote device 31
4.6.1.4 Set encryption policy at link level 31
4.6.1.5 Set authentication policy at link level 32

5 References 33

1 Introduction

The Bluetooth specification includes security features at the link level. It supports authentication (unidirectional or mutual) and encryption. These features are based on a secret link key that is shared by a pair of devices. To generate this key a pairing procedure is used when the two devices communicate for the first time.

The link level functions are defined in the Bluetooth Baseband and the Link Manager Protocol Specifications (see [1] and [2]). More comprehensive descriptions can be found in the Bluetooth ’99 conference proceedings (see [3], [4]).

The Bluetooth profiles describe how to use the Bluetooth protocols in an interoperable way. Concerning security this is done in the Generic Access Profile [5], which also defines terms used throughout this white paper. This profile specifies three security modes for a device:

- Security mode 1 (non-secure): A device will not initiate any security procedure.
- Security mode 2 (service-level enforced security): A device does not initiate security procedures before channel establishment at L2CAP level. This mode allows different and flexible access policies for applications, especially running applications with different security requirements in parallel.
- Security modes 3 (link level enforced security): A device initiates security procedures before the link set-up at the LMP level is completed.

This white paper deals with security mode 2. It describes how flexible security mechanisms can be implemented. The contents of this white paper serve only as an implementation guideline and do not represent a Bluetooth specification, since this is device specific and not required for interoperability.

Chapter 2 lists the requirements and design objectives leading to the definitions of this flexible architecture and the limitations. Chapter 3 describes a possible architecture using a central security manager.

2 Functionality & Limitations

This chapter describes the functionality of the architecture explained in Chapter 3. This can also be seen as a summary what can be built on top of security mode 2 (service-level enforced security, see [5]).

2.1 Security Levels

- It is possible to define different security levels for devices and services.
- For devices two trust levels are distinguished:
  - Trusted Device: Device with fixed relationship (paired) that is trusted and has unrestricted access to all services.
  - Untrusted Device: Device with no permanent fixed relationship (but possibly a temporary one) or device that has a fixed relationship, but is not considered as trusted. The access to services is restricted.

  A possible refinement is to set the trust level of a device specifically for services or a group of services. The interaction with the remote device does not exclude the implementation of such refined access policies.
- For services the requirement for authorisation, authentication and encryption are set independently (although some restrictions apply). The access requirements allow to define three security levels:
  - Services that require authorisation and authentication. Automatic access is only granted to trusted devices. Other devices need a manual authorisation.
  - Services that require authentication only. Authorisation is not necessary.
  - Services open to all devices; authentication is not required, no access approval required before service access is granted.
- A default security level is defined to serve the needs of legacy applications. This default policy will be used unless other settings are found in a “security” database related to a service, e.g., an internal security information database.

2.2 Flexibility and Usability

- It is possible to grant access to some services without providing access to other services (example: On a cellular phone, Service Discovery records shall be accessible, whereas dialup networking shall only be available for specific devices.)
- The security architecture supports security policies for devices with some services communicating with changing remote devices (example: File

Transfer or Business Card Exchange). Access granted to a service on such device does

  - not open up access to other services on the device.
  - not grant future access automatically or in an uncontrolled way to services on the device.

- User intervention for access to services is avoided as much as possible. It is only needed to allow devices limited access to services or for setting up trusted relationship with devices allowing unlimited access to services.
- This architecture does not deal with application level security, but such concepts are not excluded.

2.3 Implementation and Bluetooth-specific Matters

- The security architecture accounts for Bluetooth multiplexing protocols at and above L2CAP. At present, only RFCOMM is considered, as all other protocols are not Bluetooth-specific, and some have their own security features.
- The security architecture allows different protocols to enforce the security policies. For example, L2CAP will enforce the Bluetooth security policy for cordless telephony, RFCOMM will enforce the Bluetooth security policy for dialup networking, and OBEX will use its own security policy for file transfer and synchronisation.
- The architecture can completely work using security mode 2 of the Generic Access Profile. Especially since there are no changes to Baseband and LMP functions for authentication and encryption.
- Authentication and encryption are set for a physical connection (i.e., on baseband level).
- Lower layers are not aware of service/application layer security.
- The enforcement policy for authentication, authorisation or encryption might be different for client and server role. The security level of peer entities running an application needs not to be symmetric.

2.4 Risks and Limitations:

The following scenarios have been considered in identifying the limitations.

- Scenario 1: There are two Bluetooth devices (e.g., PDAs). Each device has a set of applications: calendar, file synchronization, etc. The two devices will communicate, over a Bluetooth link, to perform a certain task such as file synchronization.

- Scenario 2: There are more than two of scenario 1 devices. All devices will communicate over Bluetooth links to perform tasks that do not require security such as exchanging business cards.

- Scenario 3: A small device such as a PDA requires access, over a Bluetooth link, to infrastructure services: the Internet, e-commerce applications, corporate database, etc. Such device will be connected to a "LAN Access Point" over the BT link. The LAN Access Point will be connected to the infrastructure services via a wired or wireless LAN. This is a 3-tier configuration where tier 1 is the small device, tier 2 is the LAN Access Point, and tier 3 is the Infrastructure Services.

The Bluetooth Security architecture has the following limitations:

1. Support for legacy applications: In all scenarios, the legacy application will not make calls to the security manager. Instead a Bluetooth-aware “adapter” application is required to make security-related calls to the Bluetooth security manager on behalf of the legacy application.

2. Only a device is authenticated and not its user. If there is a need for authentication of the user, other means – e.g., application level security features – will be necessary.

3. Refer to scenario 1. There is no mechanism defined to preset authorisation per service. However, a more flexible security policy can be implemented with this architecture, without a need to change the Bluetooth protocol stack. Of course, modifications of the security manager and the registration processes would be necessary.

4. The approach only allows access control at connection set-up. The access check can be asymmetric, but once a connection is established, data flow is in principle bi-directional. It is not possible within the scope of this architecture to enforce unidirectional traffic.

5. Support for the 3-tier configuration in scenario 3: The security architecture presented in this paper is built upon the Bluetooth baseband security procedures that addresses the BT link security and mutual device authentication at each end of the link. To address the end-to-end security issue present in cases like in scenario 3, this paper assumes the existence of a “higher-level” end-to-end security solution which may utilize the Bluetooth security architecture presented for accessing devices and services directly present at the two ends of a Bluetooth link. This higher-level security solution is outside the scope of this paper.

3 Security Architecture to meet the Requirements

This chapter describes an approach for a flexible security architecture built on top of the link-level security features of Bluetooth.

3.1 Overview

The general architecture is shown in Figure 1. The key component is a security manager with the following tasks:

- Store security-related information on services
- Store security-related information on devices
- Answer access requests by protocol implementations or applications (access granted or refused)
- Enforce authentication and/or encryption before connecting to the application.
- Initiate or process input from an ESCE[1] (e.g., the device user) to set-up trusted relationships on device level.
- Initiate pairing and query PIN entry by the user. PIN entry might also be done by an application.

More details will be described in the following sections.

This approach is devoted to connection-oriented L2CAP channels. The sections up to 3.4 only deal with this. For connectionless L2CAP data transmission, restrictions apply, which will be discussed in Section 3.5.

The security architecture presented in this paper provides a very flexible security framework. This framework dictates when to involve a user (e.g., to provide a PIN) and what actions the underlying BT protocol layers follow to support the desired security check-ups. Within this framework, a number of realizations of the presented architecture can be instantiated, some of them simpler and some of them more advanced than the one discussed in detail in this paper, without moving outside the scope of the architecture.

[1] ESCE stands for “External Security Control Entity.” ESCE typically represents a human operating a device who decides how to proceed with security related matters, e.g., provide a PIN whenever needed, decide to create a trust relation with a device, etc. In general though, an ESCE represents an entity with the authority and knowledge to make decisions on how to proceed in a manner consistent to this security architecture. It could be a device user, or a utility application executed on behalf of the user based on preprogrammed security policies. In the latter case, this utility could reside within or outside a particular BT-enabled device. Without lack of generality, in the sequel the terms “ESCE” and “user” will be used interchangably and without any distinction.

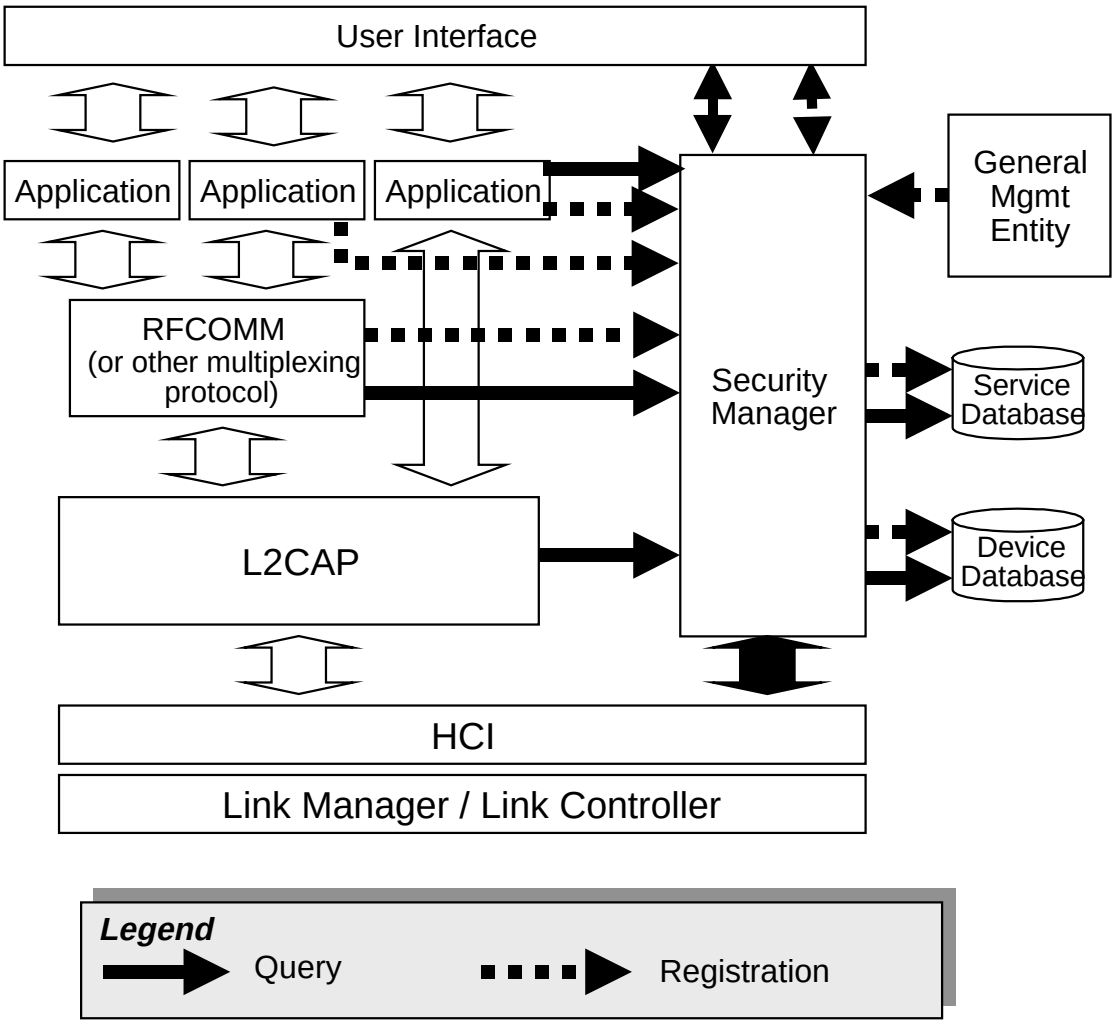

Figure 1: Security Architecture

This approach with a centralised security manager allows easy implementation of flexible access policies, because the interfaces to protocols and other entities are kept simple and are limited to query/response and registration procedures. The policies for access control are encapsulated in the security manager. Therefore, implementation of more complex policies would not affect implementation of other parts.

Implementations may decide, who performs the registration task, e.g., the application itself or a general management entity, responsible for setting the path in the protocol stack and/or registering the service at service discovery. This will be further discussed in Section 3.3.4.

3.2 Security Levels

3.2.1 Authorisation and Authentication

We distinguish between authentication and authorisation. The terms are defined as follows:

Authentication is the process of verifying 'who' is at the other end of the link. Authentication is performed for devices (BD_ADDR). In Bluetooth this is achieved by the authentication procedure based on the stored link key or by pairing (entering a PIN).

Authorisation is the process of deciding if device X is allowed to have access to service Y. This is where the concept of '**trusted'** exists. Trusted devices (authenticated and indicated as "trusted"), are allowed access to services. Untrusted or unknown devices may require authorisation based on user interaction before access to services is granted. This does not principally exclude that the authorisation might be given by an application automatically. Authorisation always includes authentication.

3.2.2 Device Trust Level

We distinguish between two different device trust levels:

Trusted Device:	The device has been previously authenticated, a link key is stored and the device is marked as "trusted" in the Device Database.
Untrusted Device:	The device has been previously authenticated, a link key is stored but the device is not marked as "trusted" in the Device Database
Unknown Device:	No security information is available for this device. This is also an untrusted device.

There will be a database table maintained in the security manager (see below). This database might be maintained for all services together (normal case referred to throughout this paper) or separately for each service or group of services.

3.2.2.1 Authenticate trusted device

The verification is done using the authentication procedure, defined in the LMP and Baseband specifications. A device is verified as trusted, if a positive authentication response is given and the trusted flag is set.

3.2.2.2 Set-up of the trusted relationship

A trusted relationship is established during the pairing procedure. This is usually performed during the bonding procedure but could be performed at connection set-up.

When an untrusted device is authorised to use a service, it is also possible to add it to the list of trusted devices during the same procedure. This of course requires an active selection by the user.

3.2.2.3 Authenticate untrusted device

Authentication of untrusted devices is done similarly as for trusted devices with the exception that the device is not marked as trusted in the internal database.

3.2.3 Security Level of Services

The security level of a service is defined by three attributes:

Authorisation Required:	Access is only granted automatically to trusted devices (i.e., devices marked as such in the device database) or untrusted devices after an authorisation procedure. Authorisation always requires authentication to verify that the remote device is the right one.
Authentication Required:	Before connecting to the application, the remote device must be authenticated
Encryption Required:	The link must be changed to encrypted mode, before access to the service is possible

This information is stored in the service database of the security manager.

If no registration has taken place, a default security level is used. This default is:

Incoming Connection:	Authorisation and Authentication required
Outgoing Connection:	Authentication required

3.3 Connection Set-up Procedure

3.3.1 General Concept

To meet different requirements on availability of services without user intervention, we must perform the authentication after determining what the security level of the requested service is. Thus, the authentication cannot be performed, when the ACL link is established. The authentication is performed, when a connection request to a service is submitted.

Figure 2 illustrates the information flow for access to a trusted service. This is intended to be an example to understand and discuss the basic principles. The details will be described further below.

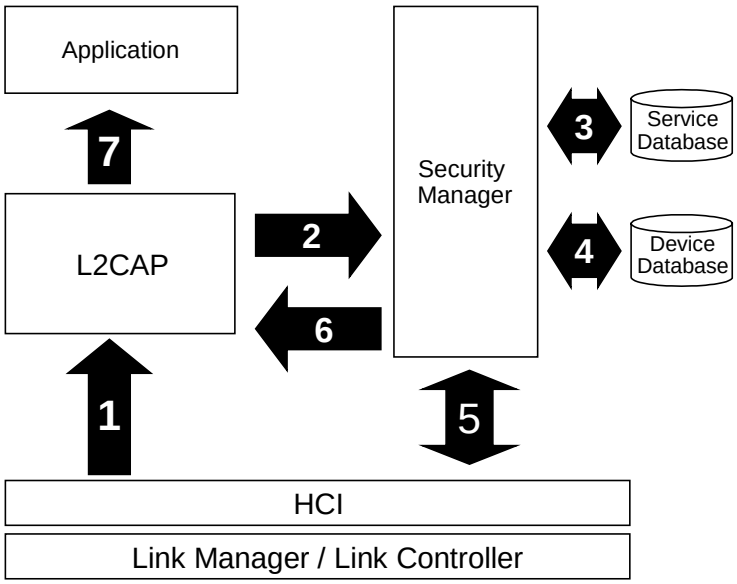

Figure 2: Information flow for access to trusted service

The following procedures are performed:

1. Connect request to L2CAP
2. L2CAP requests access from the security manager.
3. Security manager: lookup in service database
4. Security manager: lookup in device database
5. If necessary, security manager enforces authentication and encryption
6. Security manager grants access
7. L2CAP continues to set-up the connection.

Authentication can be performed in both directions, client authenticates server and vice versa.

3.3.2 Authentication on Baseband Link Set-up

Although not targeted to security mode 3 (link level enforced security, see [5]), this architecture can support this mode as well. The security manager can command the link manager to enforce authentication before accepting a BB connection using the HCI. Different modes could be implemented in parallel:

- authentication on BB connection
- authentication on connection to the applications

Clearly, they cannot run simultaneously, so if both are implemented, a decision has to be made somewhere. Before transitioning from mode 2 to mode 3, it must be ensured that untrusted devices do not get unwanted

access. To achieve this, the security manager can remove any link keys for untrusted devices stored in the radio module. The security manager can use the HCI.

3.3.3 Handling for Protocol Stack

For incoming connections, the access control procedure is described in Figure 3 for incoming connections. Access control is required at L2CAP and in some cases additionally at multiplexing protocols above (e.g., RFCOMM). When receiving a connection request, the protocol entity queries the security manager, providing any multiplexing information it received with the connect request. The security manager makes a decision whether access is granted or refused and replies to the protocol entity. If access is granted, the connection set-up procedure is continued. If access is refused, the connection is terminated.

If no access control is performed on a protocol layer, no interaction takes place with the security manager or other entities.

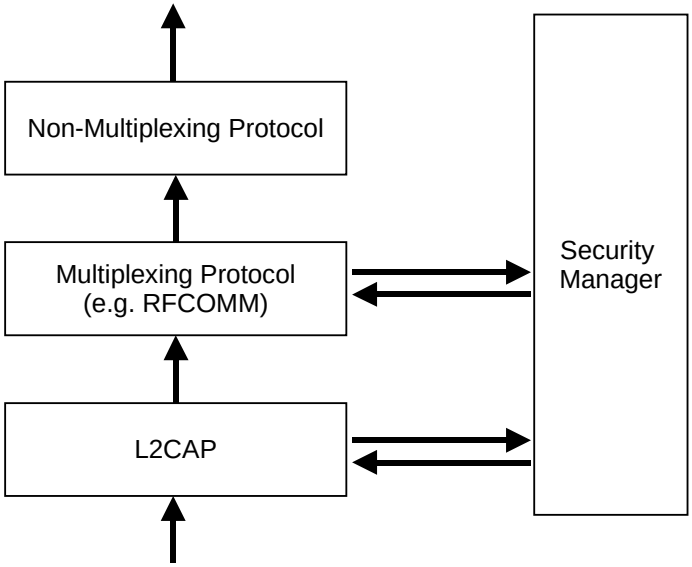

Figure 3: Behaviour of Protocols for incoming Connections

In this model we have two queries (e.g., function calls) to the security manager. The security manager should be able to store information on existing authentications . This avoids multiple authentication procedures on LMP level (i.e., over the air) within the same session.

Thus, RFCOMM will do a policy check call to security manager. This will require an additional function call but not necessarily an additional authentication.

For outgoing connections, a security check might also be required to achieve mutual authentication (authorisation is probably not useful here in most

cases). A similar procedure is carried out. The most elegant way is of course, if the applications submit requests to the security manager directly. If this is not possible (e.g., legacy applications), queries to the security manager are submitted by all multiplexing protocols from top to bottom, see Figure 4.

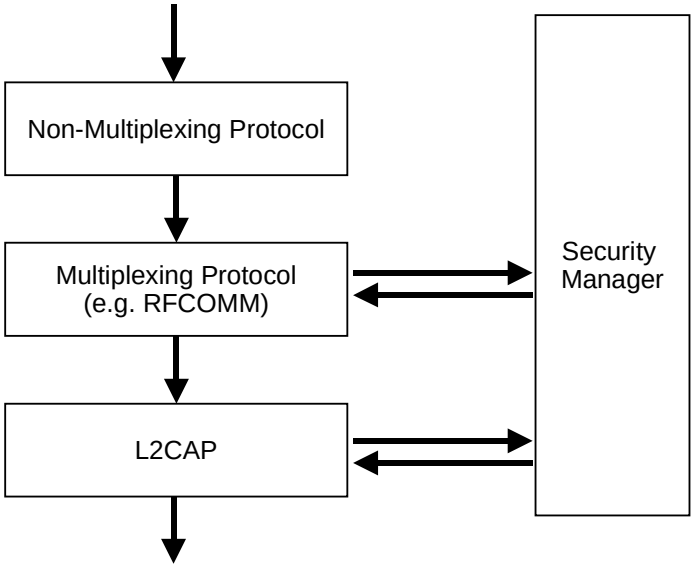

Figure 4: Behaviour of Protocols for outgoing Connections

3.3.4 Registration Procedures

As mentioned in section 3.1, the security manager maintains security information for services in security databases, the implementation of which is outside the scope of this paper. Applications must register with the security manager before becoming accessible, see Figure 5.

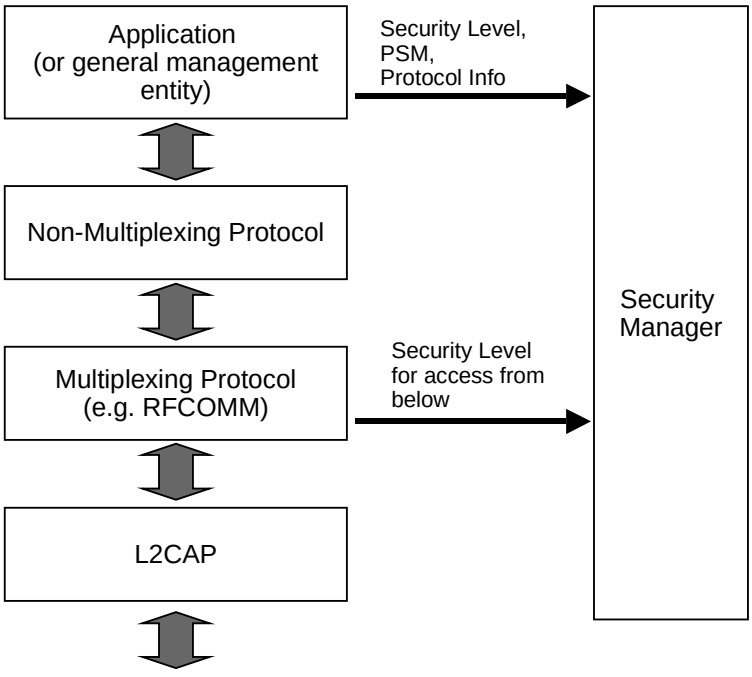

Figure 5: Registration Processes

Applications or their security delegate must provide their security level and multiplexing information (This is somewhat similar to the information registered to service discovery. The latter is required for the security manager to make a decision on a request submitted by a protocol entity as this entity will not know about the final application in most cases).

Multiplexing protocol implementations performing queries at the security manager should register the policy for accessing them from below.

Both registrations can also be done by the entity that is responsible for setting the path in the BT protocol stack. It is implementation-dependent, which entity does the registration (note: combining the service and security registration processes is not excluded).

If no registration has taken place, the security manager will assume the default security level to make a decision on access see Section 3.2.3.

L2CAP does not require registration here. It is the first multiplexing protocol in the Bluetooth stack and there will be a query for every connection request.

3.3.5External Key Management

This architecture does not exclude use of external key management procedures. Key management applications can distribute PINs or the link keys directly.[2] In this case though, one needs to proceed with caution to maintain proper interoperability of the BT-enabled devices.

3.4 Access Control Procedures

This section gives an example how the access control can be handled in the security manager. Other solutions with the same functionality and/or the same security level are possible.

[2] It is not possible to provide the encrpytion keys from outside the module.

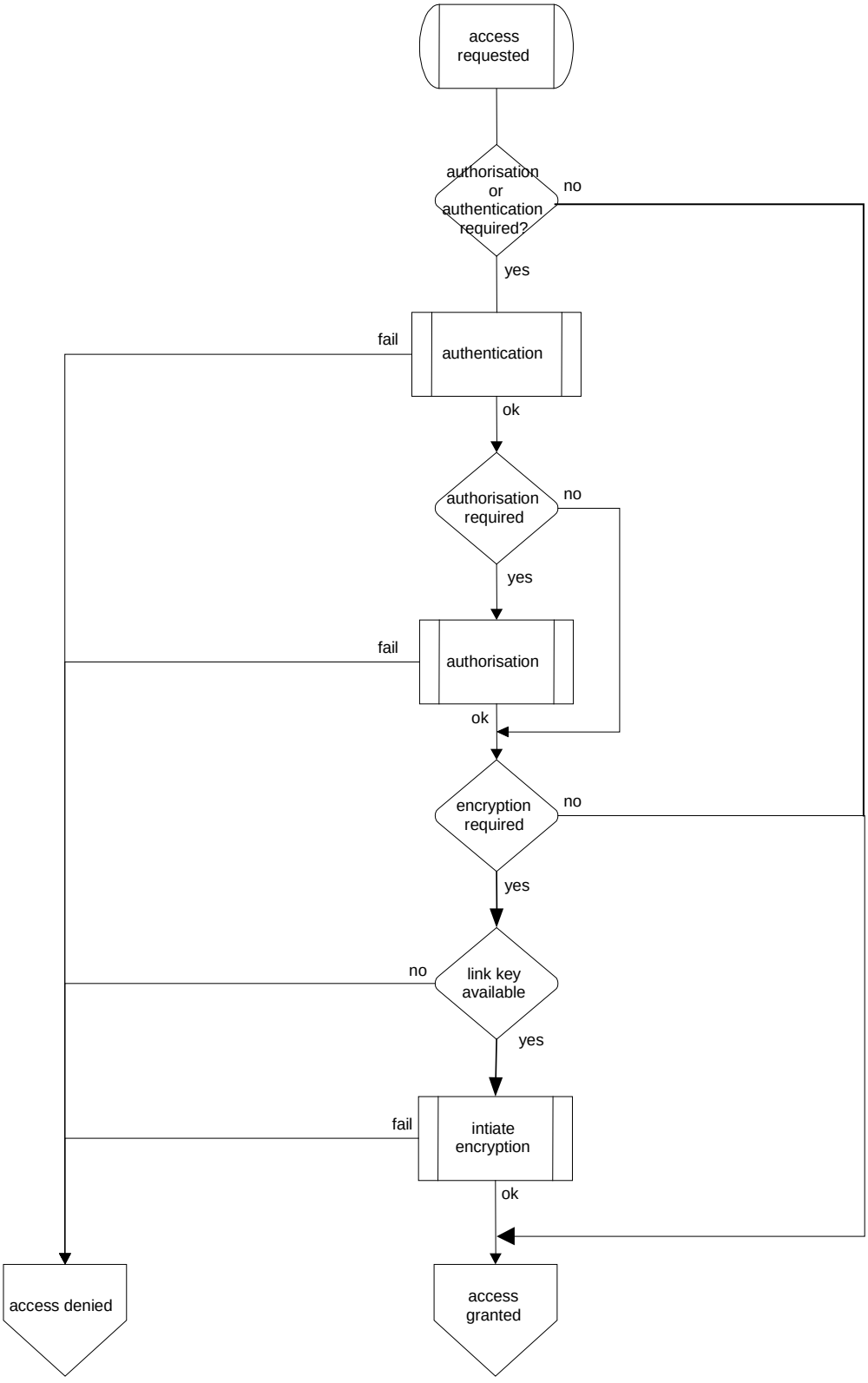

Figure 6: Example Flow Chart for Access Check by the security manager

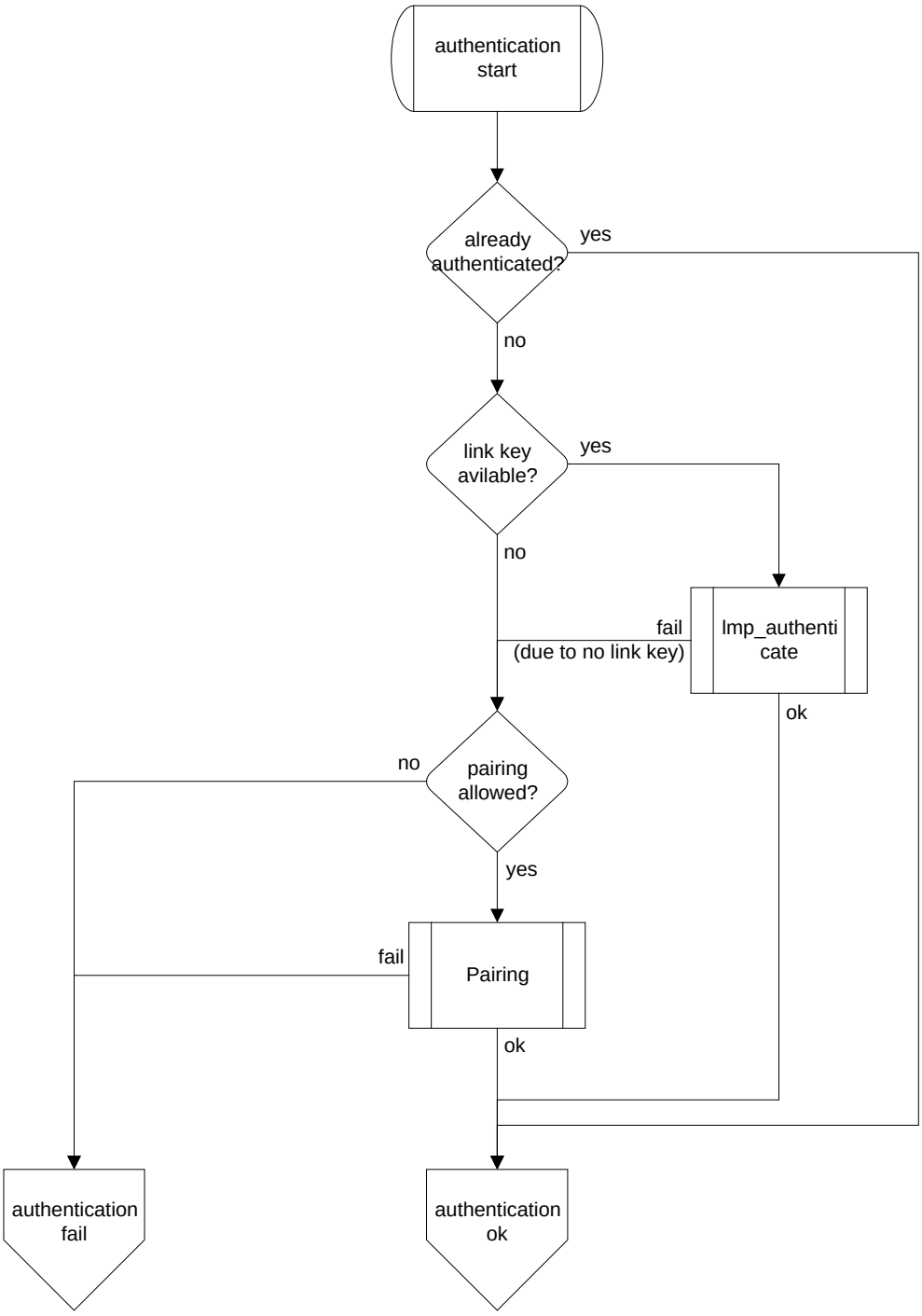

Figure 7: Example Flow Chart for Authentication Procedure

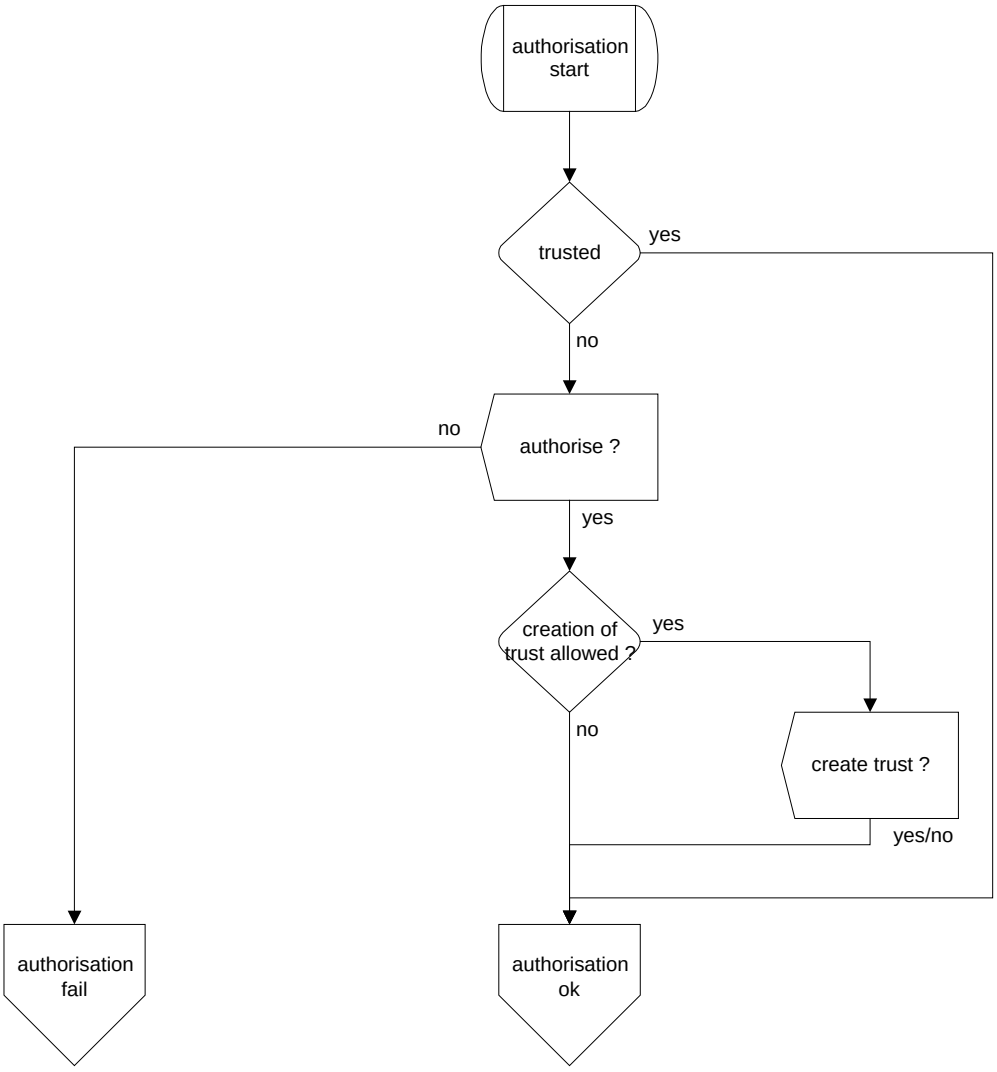

Figure 8: Example Flow Chart for Authorisation Procedure

3.5 Connectionless L2CAP

As the security check is performed at a connection request to L2CAP to set-up a connection to the next higher protocol or application, the security check of connectionless data packets cannot be performed in that way. It is not practical to perform a security check on each single connectionless data packet. Therefore, a general policy of handling connectionless packets has to be made at L2CAP level.

L2CAP offers the possibility to block connectionless traffic. This block can be done for a single protocol (PSM) on top of L2CAP, a list or all protocols. The same choices are possible for enabling connectionless traffic.

The security manager should check, whether there is any service in the service data base that does not permit connectionless data packets. The security manager will then initiate the enabling/disabling accordingly.

If connectionless packets are passed, there will be no security check. It is then up to the protocol above L2CAP to make sure that no unacceptable security problem occurs. It will always be known, whether the data came in via connectionless or connection-oriented mode, but for connectionless packets the originiator is not known or verifiable.

4 Interfaces and Functions of the Security Manager

This chapter describes a possible set of interfaces and functions for the security manager. The interactions are modelled as function calls. This chapter is meant as an implementation example. Clearly, the internal interfaces have no impact on interoperability with remote devices.

4.1 Databases

The security manager that implements the security architecture in this paper has to maintain several databases (or in general “information lists”).

For the database tables the following abbreviations are used:

- "M" for mandatory to support
- "O" for optional to support
- "C" for conditional to support

The statements mandatory, optional or conditional are relative to the discussed realization of the security architecture. Clearly, simpler or more elaborate realizations of the security implementation may have other mandatory, conditional, or optional entires.

4.1.1 Service Database

The service database has to maintain the following security-related entries for each service. This could be stored in non-volatile memory or the services register at start-up.

Authorisation Required:	M	Boolean
Authentication Required:	M	Boolean
Encryption Required:	M	Boolean
PSM Value	M	Uint16
Broadcasting allowed	O	Boolean
other routing information	C	Structure TBD (only when needed)

4.1.2 Device Database

The trust information has to be stored in non-volatile memory. If entries are deleted for some reason, the respective device is treated an unknown, i.e., set to the default access level as defined in 3.2.3.

BD_ADDR	M	48 Bit IEEE address

trust level	M	Trusted / untrusted
link key	M	Bit field (length up to 128 bit)
device name	O	String (can be used to avoid name request)

Note: It would also possible to store key information in a different way.

4.1.3 Temporary Storage

There are some data that should be stored to reduce overhead on the air interface. For each Baseband link:

- Authentication status
- Encryption status

4.2 Interface to L2CAP

L2CAP asks the security manager for access rights to a service on incoming and outgoing connection requests. There is no registration procedure since L2CAP is mandatory in Bluetooth protocol stacks.

access := SEC_accessRequest (
ProtocolIdentification,
ChannelIdentification,
BD_ADDR,
ConnectionHandle
IncomingConnection
);

Parameter	I/O	Type	Content
access	ret[3]	boolean	true = granted false = denied
ProtocolIdentification	in	uint32	Number assigned to identify, which protocol has submitted the query The value is set to zero in case of L2CAP.
PSM	in	uint32	The channel is identified by the PSM value of next protocol in

[3] "ret" stands for the value "returned" by a function call; "in" stands for the "input" parameters to a function call; "out" stands for additional parameters that are "outputed" when the function call returns.

			stack
BD_ADDR	in	48 bit	48 bit address of the remote device
ConnectionHandle[4]	in	uint16	connection handle (on HCI level) associated with ACL link to remote device
IncomingConnection	in	boolean	true = incoming connection false = outgoing connection

4.3 Interface to other multiplexing Protocols

Other multiplexing protocols (e.g., RFCOMM) that need to make decisions on access to services query the security manager in a similar way as L2CAP. There is an additional registration procedure, which allows to set the access policy for connection to the multiplexing protocol itself.

access := SEC_accessRequestMultiplexingProtocol (
ProtocolIdentification,
ChannelIdentification,
BD_ADDR,
ConnectionHandle
IncomingConnection);

Parameter	I/O	Type	Content
access	ret	boolean	true = granted false = denied
ProtocolIdentification	in	uint32	Number assigned to identify, which protocol has submitted the query
ChannelIdentification	in	uint32	Channel ID (or whatever is used in that protocol), where a decision of an access policy is based on; for RFCOMM: DLCI
BD_ADDR	in	48 bit	48 bit address of the remotes device
ConnectionHandle[5]	in	uint16	connection handle (on HCI level) associated with ACL link

[4] The information is redundant to the Bluetooth device address. However, the connection handle is needed to initiate authentication and encryption via the HCI.

			to remote device
IncomingConnection	in	boolean	true = incoming connection false = outgoing connection

4.4 Interface to ESCE (e.g., UI)

The architecture includes user interaction for authorisation purposes. This includes access permission to services and setting up a trusted relationship to a remote device.

access = SEC_authorisationRequest (
ServiceName,
DeviceName,
&FutureTrustedRelationship = false
);

The security manager calls the ESCE (e.g., the user interface); incoming parameters are the information submitted with the request, outgoing parameters hold the response.

Parameter	I/O	Type	Content
ServiceName	in	String	human readable name of the application (from registration of application)
DeviceName	in	String	human readable name of the device (retrieved using the name request or from internal database)
FutureTrustedRelationship	out	Boolean	If this value is true, the remote device will be marked as trusted. Default value is false.

If a PIN is requested by the security manager, the following call to the ESCE can be used. The PIN entry can also be requested directly from the link manager (then the security manager requests authentication, and the link manager performs the necessary actions if no valid link is available.

SEC_PinRequest (

[5] See footnote in previous section.

BD_ADDR,
Name,
PIN
);

Parameter	I/O	Type	Content
BD_ADDR	in	48 Bit	48 bit address of the remote device
Name	in	String	Bluetooth device name (human readable name)
PIN	out	String	Bit field, length < 16 bytes

In case the the ESCE wants the security manager to create a trusted relationship outside of other procedures, a simple command may be used:

SEC_createTrustedRelationship (
BD_ADDR
);

The ESCE (e.g., user interface) calls the security manager.

Parameter	I/O	Type	Content
BD_ADDR	in	48 bit	48 bit address of the remote device

4.5 Registration Procedures

There are certain registration procedures necessary:

- services with their security level and protocol stack information
- multiplexing protocols above L2CAP

This registration can be done by the entity that is responsible for setting the path in the BT protocol stack. It is implementation-dependent, which entity does the registration. Without registration the default settings apply.

SEC_registerApplication (
Name,
SecurityLevel,
PSM,
ProtocolIdentification,

ChannelIdentification
);

Parameter	I/O	Type	Content
Name	in	string	human readable name of the application (intended for user queries)
SecurityLevel	in	uint16	Bit 0–2 incoming connection: bit 0 = authorisation required bit 1 = authentication required bit 2 = encryption required Bit 3–5 outgoing connection: bit 3 = authorisation required bit 4 = authentication required bit 5 = encryption required Bit 6 = reception of connectionless packets allowed
PSM	in	uint16	PSM value used at L2CAP level
ProtocolIdentification	in	uint32	Number assigned to identify, which protocol has to make the decision for access. Zero = decision at L2CAP
ChannelIdentification	in	uint32	Channel ID (or other appropriate multiplexing identifier), where a decision of an access policy is based on; for RFCOMM: DLCI In case of *ProtocolIdentification =0*, this value has to be ignored.

SEC_registerMultiplexingProtocol (
ProtocolIdentification
LowerProtocol*,*
LowerChannel,
Security Level
);

Parameter	I/O	Type	Content
ProtocolIdentification	in	uint32	Number assigned to identify, which protocol is registered

LowerProtocol	in	uint32	ProtocolIdentification of the next lower protocol
LowerChannel	in	uint32	ChannelIdentification used at the lower layer
Security Level	in	uint16	Bit 0–2 incoming connection: bit 0 = authorisation required bit 1 = authentication required bit 2 = encryption required Bit 3–5 outgoing connection: bit 3 = authorisation required bit 4 = authentication required bit 5 = encryption required Bit 6 = reception of connectionless packets allowed

4.6 Interface to HCI / Link Manager

4.6.1.1Authentication request

For requesting an authentication of a remote device, the HCI_Authentication_Requested command and as an answer the Authentication Complete event are used which are shown below. For further details please refer to 4.5.15 and 5.2.6 of [6]

Command	OCF	Command Parameters	Return Parameters
HCI_Authentication_ Requested	0x0011	Connection_Handle	

Event	Event Code	Event Parameters
Authentication Complete	0x06	Status, Connection_Handle

4.6.1.2 Encryption control

For encryption control, the HCI_Set_Connection_Encryption command and as an answer the Encryption Change event are used to enable and disable the link level encryption. For further details please refer to 4.5.16 and 5.2.8 of [6].

Command	OCF	Command Parameters	Return Parameters
HCI_Set_ Connection_ Encryption	0x0013	Connection_Handle, Encryption_ Enable	

Event	Event Code	Event Parameters
Encryption Change	0x08	Status, Connection_Handle, Encryption_Enable

4.6.1.3 Name request to remote device

For a name request to a remote device, the HCI_Remote_Name_Request command and as an answer the Remote Name Request Complete event are used which are shown below. For further details please refer to 4.5.19 and 5.2.7 of [6].

Command	OCF	Command Parameters	Return Parameters
HCI_Remote_Name_ Request	0x0019	BD_ADDR, Page_Scan_Repetition_ Mode, Page_Scan_Mode, Clock_Offset	

Event	Event Code	Event Parameters
Remote Name Request Complete	0x07	Status, BD_ADDR, Remote_Name

4.6.1.4 Set encryption policy at link level

The general encryption policy at link level can be set by the HCI_Write_Encryption_Mode command which will be answered by the Command Complete event, both shown below. The Encryption Mode parameter controls if the Bluetooth radio will require encryption at link level for each connection with other Bluetooth radios. For further details please refer to 4.7.25 and 5.2.14 of [6].

Command	OCF	Command Parameters	Return Parameters
HCI_Write_ Encryption _ Mode	0x0022	Encryption_ Mode	Status

Event	Event Code	Event Parameters
Command Complete	0x0E	Num_HCI_Command_Packets, Command_Opcode, Return_Parameters

4.6.1.5 Set authentication policy at link level

The general authentication policy at link level can be set the HCI_Write_Authentication_Enable command which is shown below. The Authentication_Enable parameter controls if the Bluetooth radio will require authentication at link level for each connection with other Bluetooth radios. As for the HCI_Write_Encryption_Mode answer, the Command Complete event is used. For further details please refer to 4.7.23 and 5.2.14 of [6].

Command	OCF	Command Parameters	Return Parameters
HCI_Write_ Authentication _Enable	0x0020	Authentication_Enable	Status

5 References

[1] Bluetooth Baseband. Bluetooth Specification Section B:14

[2] Bluetooth Link Manager Protocol, Bluetooth Specification Section C

[3] Thomas Müller: Bluetooth Security. Proceedings Bluetooth'99, London, June 1999

[4] Joakim Persson: Bluetooth Baseband Security Concept. Proceedings Bluetooth'99, London, June 1999

[5] Generic Access Profile, Bluetooth Specification Section

[6] Bluetooth Host Controller Interface, Bluetooth Specification Section H1