SECURITY SERVICE

INDEX

INTRODUCTION 3

COMMANDS 4

XPEAK_COMMAND_CREATE_PIN_BLOCK (v.0706) 5
XPEAK_COMMAND_DECRYPT (v.0706) 7
XPEAK_COMMAND_DELETE_KEYS (v.0706) 8
XPEAK_COMMAND_ENCRYPT (v.0706) 9
XPEAK_COMMAND_GENERATE_MAC (v.0706) 10
XPEAK_COMMAND_GET_CAPABILITIES (v.0706) 11
XPEAK_COMMAND_GET_KEYS_LOADED (v.0706) 14
XPEAK_COMMAND_GET_SERIAL_NUMBER (v.0706) 15
XPEAK_COMMAND_LOAD_KEY (v.0706) 16
XPEAK_COMMAND_LOAD_MANUAL_KEY_END (v.0706) 19
XPEAK_COMMAND_LOAD_MANUAL_KEY_START (v.0706) 21
XPEAK_COMMAND_VALIDATE_PIN (v.0706) 24

INTRODUCTION

This documentation details the specific set of commands for security devices. These, along with ***common commands*** conform the complete set of commands available for security devices. Sometimes, certain ***common commands*** can be overwritten within a specific service, because they change their behavior. In the case of **Security Service**, following commands are overwritten:

- XPEAK_COMMAND_GET_CAPABILITIES

All commands described here meet the ***Xpeak*** specification General Message Format

COMMANDS

XPEAK_COMMAND_CREATE_PIN_BLOCK (0x706010E)

Version: 0706

Description:

Generates the PIN block using the specified key, the typed PIN and the application data.

There are two different ways to insert the PIN:

Case	Description
1	The PIN is inserted through a XPEAK_SERVICE_TYPE_KEYBOARD service, using the command XPEAK_COMMAND_READ of the service KEYBOARD with parameter ***Mode*** equals XPEAK_KEYBOARD_READ_MODE_PIN (0x7060104). This option is only possible if the capability ***PinEntryViaKeyboardService*** from the XPEAK_COMMAND_GET_CAPABILITIES command is ***true***.
2	The PIN is inserted without the intervention of any XPEAK_SERVICE_TYPE_KEYBOARD service. During the execution of this command (XPEAK_COMMAND_CREATE_PIN_BLOCK), the user should enter the PIN with which the service will generate the PIN block. The value of the capability ***PinEntryViaKeyboardService*** from the XPEAK_COMMAND_GET_CAPABILITIES command should be ***false*** in this case.

Parameters:

- **`String Key`**
  Name of the key (with XPEAK_SECURITY_USE_PIN (0x7060125) use) used to encrypt the PIN block. If the key has not XPEAK_SECURITY_USE_PIN (0x7060125) use, the error XPEAK_RESULT_INVALID_KEY (0x7060119) will be returned.

  If no encryption is required, this parameter should be the empty array ("").

- **`String Data`**
  Data used to generate the PIN block. The String is the hexadecimal representation of the data. Each byte is represented by two characters of the String value.

- **`String PaddingChar`**
  Contains the padding char used in some PIN block formats. The length of this String should be 1 or 0 if no padding char is needed.

- **`int Format`**
  Indicates the PIN format. This format has to be one of the ***PinFormats*** supported by the device, according to the XPEAK_COMMAND_GET_CAPABILITIES command.

Result:

- **`String PinBlock`**
  The resulting data. The length of the PIN block will be 8 Bytes. Each byte is represented by two characters of the String value.

- **`int Result`**

  Apart from the common values, this command could return as result:

Result	Description
XPEAK_RESULT_INVALID_KEY (0x7060119)	The ***Key*** has not the properly use for this operation.
XPEAK_RESULT_KEY_NOT_FOUND (0x706011A)	No key with the specified name has been loaded in the security service.
XPEAK_RESULT_NO_TYPED_PIN (0x7060135)	The PIN has not been typed.
XPEAK_RESULT_WRONG_TYPED_PIN (0x7060136)	The length of the typed PIN is wrong or any of the pressed keys is not allowed.

XPEAK_COMMAND_DECRYPT (0x7060111)

Version: 0706

Description:

Decrypts ***Data*** with the specified ***Key*** using any of the encryption algorithms supported by the device.

Parameters:

- **String Key**
  Indicates the name of the key used to decrypt. This key must have XPEAK_SECURITY_USE_CRYPT (0x7060123) use, otherwise the error XPEAK_RESULT_INVALID_KEY (0x7060119) will be returned.

- **String Data**
  Contains the data to be decrypted. The String is the hexadecimal representation of the data, which should be a multiple of 8 bytes length. Each byte is represented by two characters of the String value.

- **String InitializationVector**
  The initialization vector is used in the CBC and CFB algorithms. If this parameter is the empty string (""), the value 000000000000000 will be used as initialization vector.

- **int Algorithm**
  Contains the algorithm used to decrypt. This algorithm must be one of the ***CryptAlgorithms*** supported by the device and specified in the command XPEAK_COMMAND_GET_CAPABILITIES.

Result:

- **String CryptedData**
  The resulting decrypted data that should have the same number of bytes as the parameter ***Data***. Each byte is represented by two characters of the String value.

- **int Result**
  Apart from the common values, this command could return as result:

Result	Description
XPEAK_RESULT_INVALID_KEY (0x7060119)	The key has not the properly use for this operation.
XPEAK_RESULT_KEY_NOT_FOUND (0x706011A)	No key with the specified name has been loaded in the security service.

XPEAK_COMMAND_DELETE_KEYS (0x7060113)

Version: 0706

Description:

This command deletes the specified keys.

Some devices have a special ***Master*** key loaded by the manufacturer that can not be deleted. This kind of keys will be indicated by the ***PreLoaded*** field of the XPEAK_COMMAND_GET_KEYS_LOADED command.

Only the keys loaded via the command XPEAK_COMMAND_LOAD_KEY and keys loaded manually via the commands XPEAK_COMMAND_LOAD_MANUAL_KEY_START and XPEAK_COMMAND_LOAD_MANUAL_KEY_END can be deleted.

Parameters:

- **`String[] Keys`**
  Names of the keys to delete. An empty array will indicate that all the keys should be deleted (except the pre-loaded keys).

Result:

- **`int Result`**
  Apart from the common values, this command could return as result:

Result	Description
XPEAK_RESULT_KEY_NOT_FOUND (0x706011A)	Any of the key names specified does not exist in the SECURITY service.

XPEAK_COMMAND_ENCRYPT (0x7060110)

Version: 0706

Description:

Encrypts ***Data*** with the specified ***Key*** using any of the encryption algorithms supported by the device.

Parameters:

- **String Key**
  Indicates the name of the key used to encrypt. This key must have XPEAK_SECURITY_USE_CRYPT (0x7060123) use, otherwise the error XPEAK_RESULT_INVALID_KEY (0x7060119) will be returned.

- **String Data**
  Contains the data to be encrypted. The String is the hexadecimal representation of the data, which should be a multiple of 8 bytes length. Each byte is represented by two characters of the String value.

- **String InitializationVector**
  The initialization vector is used in the CBC and CFB algorithms. If this parameter is the empty String (""), the value 000000000000000 will be used as initialization vector.

- **int Algorithm**
  Contains the algorithm used to encrypt. This algorithm must be one of the ***CryptAlgorithms*** supported by the device and specified in the command XPEAK_COMMAND_GET_CAPABILITIES.

Result:

- **String CryptedData**
  The resulting encrypted data that should have the same number of bytes as the parameter ***Data***. Each byte is represented by two characters of the String value.

- **int Result**
  Apart from the common values, this command could return as result:

Result	Description
XPEAK_RESULT_INVALID_KEY (0x7060119)	The ***Key*** has not the properly use for this operation.
XPEAK_RESULT_KEY_NOT_FOUND (0x706011A)	No key with the specified name has been loaded in the security service.

XPEAK_COMMAND_GENERATE_MAC (0x7060114)

Version: 0706

Description:

Generates the CBC MAC of the data using the specified key, the initialization vector and any of the MAC algorithms supported by the device.

Parameters:

- **`String Key`**
  Indicates the name of the key used during the encryption algorithm. This key has to be imported with the XPEAK_SECURITY_USE_MAC (0x7060124) use. Otherwise, the error XPEAK_RESULT_INVALID_KEY (0x7060119) will be returned.

- **`String InitializationVector`**
  The initialization vector used during the MAC algorithm. If this parameter is an empty string (""), the value 000000000000000 will be used as initialization vector.

- **`String Data`**
  Contains the data for the MAC generation. The String is the hexadecimal representation of the data, which should be a multiple of 8 bytes length. Each byte is represented by two characters of the String value.

- **`int Algorithm`**
  Contains the algorithm used for the MAC generation. This algorithm has to be one of the ***`MacAlgorithms`*** supported by the device andspecified in the command XPEAK_COMMAND_GET_CAPABILITIES.

Result:

- **`String CryptedData`**
  The resulting data. Each byte is represented by two characters of the String value.

- **`int Result`**
  Apart from the common values, this command could return as result:

Result	Description
XPEAK_RESULT_INVALID_KEY (0x7060119)	Error due to the key has not the properly uses for this operation.
XPEAK_RESULT_KEY_NOT_FOUND (0x706011A)	No keys with the specified name has been loaded in the security service.

XPEAK_COMMAND_GET_CAPABILITIES (0x7060017)

Version: 0706

Description:

Returns the device capabilities. Depending on them, the application behavior should be different and should be adapted to the peripheral characteristics.

Result:

- **`boolean PinEntryViaKeyboardService`**
  If ***`true`***, the PIN is inserted through a XPEAK_SERVICE_TYPE_KEYBOARD service, using the command XPEAK_COMMAND_READ of the service KEYBOARD, with ***`Mode`*** equals XPEAK_KEYBOARD_READ_MODE_PIN (0x7060104).

  If ***`false`***, the PIN is inserted without the intervention of any XPEAK_SERVICE_TYPE_KEYBOARD service. The PIN will be typed during the execution of the command where the PIN is used. See commands XPEAK_COMMAND_CREATE_PIN_BLOCK and XPEAK_COMMAND_VALIDATE_PIN.

- **`int[] SupportedKeyLengths`**
  This array contains the key lengths supported by the device. The lengths are indicated in number of bytes. The length of the keys loaded via the command XPEAK_COMMAND_LOAD_KEY should be any of the lengths indicated in this parameter.

- **`boolean CanLoadPlainTextKeys`**
  If this result is ***`true`***, it is possible to load plain text keys through the command XPEAK_COMMAND_LOAD_KEY (i.e. without a Key Encryption Key). Otherwise (if ***`false`***), the only way to load a plain text key is loading it manually, since the SECURITY.XPEAK_COMMAND_LOAD_KEY command would require the use of a Key Encryption Key.

- **`boolean CanLoadKeysWithSeveralUses`**
  This capability indicates whether it is possible to load new keys with several uses (***`true`***) or not.

- **`boolean CanLoadKeysManually`**
  If ***`false`***, the service does not support loading keys manually. In this case, the only way to load a new key is through the command XPEAK_COMMAND_LOAD_KEY. If ***`true`***, the keyboard allows the possibility of loading keys manually. In this case, the commands XPEAK_COMMAND_LOAD_MANUAL_KEY_START and XPEAK_COMMAND_LOAD_MANUAL_KEY_END will be available.

  The value of the new key will be typed through a XPEAK_SERVICE_TYPE_KEYBOARD service, using the command XPEAK_COMMAND_READ with parameter ***`Mode`*** equals XPEAK_KEYBOARD_READ_MODE_MANUAL_KEY (0x7060105). The sequence to load a key manually should be:

- Execute the command XPEAK_COMMAND_LOAD_MANUAL_KEY_START.
- Execute the command XPEAK_COMMAND_READ of the KEYBOARD service.
- Execute the command XPEAK_COMMAND_LOAD_MANUAL_KEY_END.

The loaded key will be a key with the ***use*** XPEAK_SECURITY_USE_KEY_ENCRYPTION_KEY (0x7060122).

- **`int MaximumKeysNumber`**
  Specifies the maximum number of keys that can be loaded.

- **`int[] CryptAlgorithms`**
  Contains the encryption algorithms supported by the device. Possible values are:

Algorithm	Description
XPEAK_SECURITY_CRYPT_ALGORITHM_DESECB (0x706011B)	Electronic Code Book
XPEAK_SECURITY_CRYPT_ALGORITHM_DESCBC (0x706011C)	Cipher Block Chaining
XPEAK_SECURITY_CRYPT_ALGORITHM_DESCFB (0x706011D)	Cipher Feed Back
XPEAK_SECURITY_CRYPT_ALGORITHM_TDESECB (0x706011E)	Triple DES with Electronic Code Book
XPEAK_SECURITY_CRYPT_ALGORITHM_TDESCBC (0x706011F)	Triple DES with Cipher Block Chaining
XPEAK_SECURITY_CRYPT_ALGORITHM_TDESCFB (0x7060120)	Triple DES with Cipher Feed Back

- **`int[] MacAlgorithms`**
  Contains algorithms supported by the device for the MAC generation. Possible values are:
  - XPEAK_SECURITY_MAC_ANSI_X9_9 (0x7060126)
  - XPEAK_SECURITY_MAC_ANSI_X9_19 (0x7060127)

- **`int[] PinFormats`**
  Contains the PIN formats supported by the device. This formats are only used as parameter for the commnad XPEAK_COMMAND_CREATE_PIN_BLOCK. Possible values are:

Format	Description
XPEAK_SECURITY_PIN_FORMAT_IBM3624 (0x7060128)	PIN left justified, filled with padding characters, PIN length 4-16 digits. The Padding Character is a Hexadecimal Digit in the range 0x00 to 0x0F.
XPEAK_SECURITY_PIN_FORMAT_ANSI (0x7060129)	PIN is preceded by 0x00 and the length of the PIN (0x04 to 0x0C), filled with padding character 0x0F to the right, PIN length 4-12 digits, XORed with PAN (Primary Account Number, minimum 12 digits without check number).
XPEAK_SECURITY_PIN_FORMAT_ISO0 (0x706012A)	PIN is preceded by 0x00 and the length of the PIN (0x04 to 0x0C), filled with padding character 0x0F to the right, PIN length 4-12 digits, XORed with PAN (Primary Account Number, no minimum length specified, missing digits are filled with 0x00)
XPEAK_SECURITY_PIN_FORMAT_ISO1 (0x706012B)	PIN is preceded by 0x01 and the length of the PIN (0x04 to 0x0C), padding characters are taken from a transaction field (10 digits).

Format	Description
XPEAK_SECURITY_PIN_FORMAT_ECI2 (0x706012C)	Similar to IBM3624, PIN only 4 digits ECI3 PIN is preceded by the length (digit), PIN length 4-6 digits, the padding character can range from X'0' through X'F'.
XPEAK_SECURITY_PIN_FORMAT_VISA (0x706012D)	PIN is preceded by the length (digit), PIN length 4-6 digits. If the PIN length is less than six digits the PIN is filled with X'0' to the length of six, the padding character can range from X '0' through X '9' (This format is also referred to as VISA2).
XPEAK_SECURITY_PIN_FORMAT_DIEBOLD (0x706012E)	PIN is padded with the padding character and may be not encrypted, single encrypted or double encrypted.
XPEAK_SECURITY_PIN_FORMAT_DIEBOLDCO (0x706012F)	PIN with the length of 4 to 12 digits, each one with a value of X'0' to X'9', is preceded by the one-digit coordination number with a value from X'0' to X'F', padded with the padding character with a value from X'0' to X'F' and may be not encrypted, single encrypted or double encrypted.
XPEAK_SECURITY_PIN_FORMAT_VISA3 (0x7060130)	PIN with the length of 4 to 12 digits, each one with a value of X'0' to X'9', is followed by a delimiter with the value of X'F' and then padded by the padding character with a value between X'0' to X'F'.
XPEAK_SECURITY_PIN_FORMAT_BANKSYS (0x7060131)	PIN is encrypted and formatted according to the Banksys Pin Block specifications.
XPEAK_SECURITY_PIN_FORMAT_EMV (0x7060132)	The PIN block is constructed as follows: PIN is preceded by 0x02 and the length of the PIN (0x04 to 0x0C), filled with padding character 0x0F to the right, formatted up to 248 bytes of other data as defined within the EMV 4.0 specifications and finally encrypted with an RSA key.
XPEAK_SECURITY_PIN_FORMAT_ISO3 (0x7060133)	PIN is preceded by 0x03 and the length of the PIN (0x04 to 0x0C), padding characters sequentially or randomly chosen, XORed with digits from PAN.

XPEAK_COMMAND_GET_KEYS_LOADED (0x7060115)

Version: 0706

Description:

Returns the information of all the loaded keys.

Result:

- **`Struct[] KeysLoaded`**
  Contains the information of all the keys of the XPEAK_SERVICE_TYPE_SECURITY service.
  - **`String Name`**
    Name of the key.
  - **`int[] Uses`**
    Array with the uses of the key. If the field ***`CanLoadKeysWithSeveralUses`*** returned by the command XPEAK_COMMAND_GET_CAPABILITIES is ***`false`***, each key should have only one use. Possible uses are defined in the field ***`Uses`*** of the command XPEAK_COMMAND_LOAD_KEY.
  - **`boolean LoadCompleted`**
    Indicates if all the components of the key have been loaded. If the key has just one component (only one command XPEAK_COMMAND_LOAD_KEY is needed) the value of this field will be ***`true`***. If the key is compound, this value will be ***`true`*** only when the last component is loaded.
  - **`boolean Preloaded`**
    Indicates if the key was loaded by the manufacturer (***`true`***), so it can not be deleted. This value should be ***`false`*** in all keys stored via XPEAK_COMMAND_LOAD_KEY and keys imported manually via XPEAK_COMMAND_LOAD_MANUAL_KEY_START and XPEAK_COMMAND_LOAD_MANUAL_KEY_END.

XPEAK_COMMAND_GET_SERIAL_NUMBER (0x7060116)

Version: 0706

Description:

This command returns the serial number of the device. The serial number should be an unique device number provided by the manufacturer.

Result:

- **`String SerialNumber`**
  Contains the serial number of the device.

XPEAK_COMMAND_LOAD_KEY (0x7060112)

Version: 0706

Description:

This command is used to load a new key. The new key will be stored inside the security service. The length of the new key has to be one of the supported key lengths specified in the parameter ***SupportedKeyLengths*** of the command XPEAK_COMMAND_GET_CAPABILITIES.

The keys can be stored using a key encryption key or not, depending if the device supports loading plain text keys or not. If no Key Encryption Key is used, the ***Value*** will contain the plain text value of the key. If a Key Encryption Key is used, the ***Value*** parameter should contain the value of the key encrypted with the Key Encryption Key.

The command will return the Key Check Value (KCV) of the key. This value is the result of encrypting the value 0000000000000000 with the stored key.

Parameters:

- **String Name**

  Specifies the name of the key. This name will be used to reference the key in the rest of the commands.

  If the service already contains any key with the same name, there are two posibilities:

Case	Description
1	The existing key was loaded with the field ***LastPart*** equals ***false*** indicating the key is compound by several components. In this case, the device will store the result of a XOR operation between both keys.
2	The existing key was loaded with the field ***LastPart*** equals ***true*** indicating the load is completed. If so, the service will return the error XPEAK_RESULT_KEY_ALREADY_EXIST (0x7060134). It is necessary to delete the key via the command XPEAK_COMMAND_DELETE_KEYS before loading a key with an existing name.

- **String KeyEncryptionKey**

  Indicates the name of the Key Encryption Key. If no Key Encryption key is used, the field will contain the empty String (""). If no key with the name of ***KeyEncryptionKey*** exists, the error XPEAK_RESULT_KEY_NOT_FOUND (0x706011A) will be returned. If the specified key has not the XPEAK_SECURITY_USE_KEY_ENCRYPTION_KEY (0x7060122) use, the error XPEAK_RESULT_INVALID_KEY (0x7060119) will be returned.

  If no Key Encryption Key is used, the service should support loading plain text keys. See the command XPEAK_COMMAND_GET_CAPABILITIES.

- **boolean LastPart**

  Indicates if the data is the last component of the key. It is possible to load a key with several components. This operation is made through the execution of this command several times. The

command will be executed as many times as parts compounding the key. The complete key stored by the device will be the result of XOR operation between all the components.

If a key is not compound by several components, it will be loaded in one operation. In this case this parameter should be ***true***.

- **`String Value`**
  Contains the value of the key. The length of the key should be a 8 bytes multiple (each byte is represented by two characters of the String value), and the length has to be included in the the supported key lengths specified in the field ***SupportedKeyLengths*** of the command XPEAK_COMMAND_GET_CAPABILITIES.

  If no Key Encryption Key is used, this value should be the plain text value of the key. If a Key Encryption Key is used, this field will be the plain text value of the key encrypted with the Key Encryption Key.

- **`int[] Uses`**
  Contains the uses of the new key. If the field ***CanLoadKeysWithSeveralUses*** returned by the command XPEAK_COMMAND_GET_CAPABILITIES is ***false***, each key should have only one use. In the other case, several uses can be available for the same key.

  Possible values are:

Use	Description
XPEAK_SECURITY_USE_KEY_ENCRYPTION_KEY (0x7060122)	Keys with this use are used to load new encrypted keys.
XPEAK_SECURITY_USE_CRYPT (0x7060123)	Keys with this use can be used in encryption and decryption commands: XPEAK_COMMAND_ENCRYPT and SECURITY.XPEAK_COMMAND_DECRYPT.
XPEAK_SECURITY_USE_MAC (0x7060124)	Keys with this use can be used to generate the MAC in the command XPEAK_COMMAND_GENERATE_MAC
XPEAK_SECURITY_USE_PIN (0x7060125)	Keys with this use can be used in PIN functions: XPEAK_COMMAND_CREATE_PIN_BLOCK and XPEAK_COMMAND_VALIDATE_PIN.

Result:

- **`String KeyCheckValue`**
  Returns the Key Check Value of the loaded key. The Key Check Value is the result of encrypting the data 000000000000000 with the key using the algorithm DESECB (for single keys) or TDESECB (for double and triple keys). The number of bytes returned depends on the device. Each byte is represented by two characters of the String.

- **`int Result`**
  Appart from the common values, this command could return as result:

Result	Description
XPEAK_RESULT_INVALID_KEY (0x7060119)	The key has not properly use for this operation.
XPEAK_RESULT_KEY_NOT_FOUND (0x706011A)	No keys with the specified name has been loaded in the security device.
XPEAK_RESULT_KEY_ALREADY_EXIST (0x7060134)	There is another key with the same name loaded in the device.

XPEAK_COMMAND_LOAD_MANUAL_KEY_END (0x7060118)

Version: 0706

Description:

Ends the sequence to load a key manually. The full sequence is the next one:

Step	Command
1	Execute the command XPEAK_COMMAND_LOAD_MANUAL_KEY_START.
2	Execute the command XPEAK_COMMAND_READ of the KEYBOARD service. In this step the user should type the plain text value of the key.
3	Execute the command XPEAK_COMMAND_LOAD_MANUAL_KEY_END.

If the new key is compound by several components, the load operation should be done in several steps. Each step will load a new component and the SECURITY device will store the result of a XOR operations between the components. The parameter ***LastPart*** indicates if the command is going to store the last component or not. If the last component has not been loaded yet, the field ***LoadCompleted*** of the command XPEAK_COMMAND_GET_KEYS_LOADED should be ***false***.

To load a key manually compound by several components, it is necessary to repeat the sequence related before. For example, to load manually a complex key compound by two components, the sequence will be the next one:

Step	Command
1	Execute the command XPEAK_COMMAND_LOAD_MANUAL_KEY_START. The value of the parameter ***LastPart*** will be ***false***.
2	Execute the command XPEAK_COMMAND_READ of the KEYBOARD service. In this command the user should type the value of the key.
3	Execute the command XPEAK_COMMAND_LOAD_MANUAL_KEY_END. The value of the parameter ***LastPart*** will be ***false***.
4	Execute the command XPEAK_COMMAND_LOAD_MANUAL_KEY_START. The value of the parameter ***LastPart*** will be ***true***.
5	Execute the command XPEAK_COMMAND_READ of the XPEAK_SERVICE_TYPE_KEYBOARD service. In this command the user should type the value of the key.
6	Execute the command XPEAK_COMMAND_LOAD_MANUAL_KEY_END. The value of the parameter ***LastPart*** will be ***true***.

The fields ***Name*** and ***Uses*** should be the same during all the sequence.

Result:

- **`String KeyCheckValue`**
  Returns the Key Check Value of the loaded key. The Key Check Value is the result of encrypting the data 000000000000000 with the key using the algorithm DESECB (for single keys) or TDESECB (for double or triple keys). The number of bytes returned depends on the device. Each byte is

represented by two characters of the String.

- **`int Result`**

  Apart from the common values, this command could return as result:

Result	Description
XPEAK_RESULT_INVALID_KEY (0x7060119)	Error due to the key has not the properly uses for this operation.
XPEAK_RESULT_KEY_ALREADY_EXIST (0x7060134)	There is another key with the same name loaded in the device.
XPEAK_RESULT_NO_TYPED_KEY (0x7060138)	The value of the key has not been typed.
XPEAK_RESULT_WRONG_TYPED_KEY (0x7060139)	The length of the typed key is wrong or any of the pressed keys is not allowed.
XPEAK_RESULT_LOAD_MANUAL_KEY_NOT_STARTED (0x7060137)	The command XPEAK_COMMAND_LOAD_MANUAL_KEY_START has not executed to start the loading of a key manually.

XPEAK_COMMAND_LOAD_MANUAL_KEY_START (0x7060117)

Version: 0706

Description:

Begins the sequence to load a key manually. The full sequence is the next one:

Step	Command
1	Execute the command XPEAK_COMMAND_LOAD_MANUAL_KEY_START.
2	Execute the command XPEAK_COMMAND_READ of the XPEAK_SERVICE_TYPE_KEYBOARD service. In this step the user should type the plain text value of the key.
3	Execute the command XPEAK_COMMAND_LOAD_MANUAL_KEY_END.

If the new key is compound by several components, the load operation should be done in several steps. Each step will load a new component and the SECURITY device will store the result of a XOR operations between the components. The parameter ***LastPart*** indicates if the command is going to store the last component or not. If the last component has not been loaded yet, the field ***LoadCompleted*** of the command XPEAK_COMMAND_GET_KEYS_LOADED should be ***false***.

To load a key manually compound by several components, it is necessary to repeat the sequence related before. For example, to load manually a complex key with two components, the sequence will be the next one:

Step	Command
1	Execute the command XPEAK_COMMAND_LOAD_MANUAL_KEY_START. The value of the parameter ***LastPart*** will be ***false***.
2	Execute the command XPEAK_COMMAND_READ of the KEYBOARD service. In this command the user should type the value of the key.
3	Execute the command XPEAK_COMMAND_LOAD_MANUAL_KEY_END. The value of the parameter ***LastPart*** will be ***false***.
4	Execute the command XPEAK_COMMAND_LOAD_MANUAL_KEY_START. The value of the parameter ***LastPart*** will be ***true***.
5	Execute the command XPEAK_COMMAND_READ of the XPEAK_SERVICE_TYPE_KEYBOARD service. In this command the user should type the value of the key.
6	Execute the command XPEAK_COMMAND_LOAD_MANUAL_KEY_END. The value of the parameter ***LastPart*** will be ***true***.

The fields ***Name*** and ***Uses*** should be the same during all the sequence.

Parameters:

- **`String Name`**
  Specifies the name of the key. This name will be used to reference the key in the rest of the commands.

If the service already contains any key with the same name, there are two posibilities:

Case	
1	The existing key was loaded with the field ***LastPart*** equals ***false*** indicating the key is compound by several components. In this case, the device will store the result of a XOR operation between both keys.
2	The existing key was loaded with the field ***LastPart*** equals ***true*** indicating the load is completed. In this, the service will return the error XPEAK_RESULT_KEY_ALREADY_EXIST (0x7060134). It is necessary to delete the old key via the command XPEAK_COMMAND_DELETE_KEYS before loading a new key with an existing name.

- **`boolean LastPart`**
  Indicates if the data is the last component of the key. It is possible to load a key with several parts. This operation is made through the execution of this command several times. As many as parts compound the key. The final key stored by the device will be the result of XOR operation between the parts.

  If a key is going to be loaded in one operation, one part, this parameter should be ***true***.

- **`int[] Uses`**
  Contains the uses of the new key. If the field ***CanLoadKeysWithSeveralUses*** returned by the command XPEAK_COMMAND_GET_CAPABILITIES is ***false***, each key should have only one use. In the other case, several uses can be available for the same key.

  Possible values are:

Use	Description
XPEAK_SECURITY_USE_KEY_ENCRYPTION_KEY (0x7060122)	Keys with this use are used to encrypy new keys.
XPEAK_SECURITY_USE_CRYPT (0x7060123)	Keys with this use can be used in Crypt commnads: XPEAK_COMMAND_ENCRYPT and SECURITY.XPEAK_COMMAND_DECRYPT.
XPEAK_SECURITY_USE_MAC (0x7060124)	Keys with this use can be used generating the MAC in the command XPEAK_COMMAND_GENERATE_MAC
XPEAK_SECURITY_USE_PIN (0x7060125)	Keys with this use can be used in PIN functions: XPEAK_COMMAND_CREATE_PIN_BLOCK and XPEAK_COMMAND_VALIDATE_PIN.

- **`int KeyLength`**
  Indicates length of the typed key specified in bytes. This length has to be included in the the supported key lengths specified in the field ***SupportedKeyLengths*** of the command XPEAK_COMMAND_GET_CAPABILITIES.

Result:

- **`int Result`**
  Appart from the common values, this command could return as result:

Result	Description
XPEAK_RESULT_INVALID_KEY (0x7060119)	Error due to the key has not the properly uses for this operation.
XPEAK_RESULT_KEY_ALREADY_EXIST (0x7060134)	There is another key with the same name loaded in the device.

XPEAK_COMMAND_VALIDATE_PIN (0x706010F)

Version: 0706

Description:

Validates the typed PIN and returns the result of the validation.

There are two different ways to insert the PIN:

Case	Description
1	The PIN is inserted through a XPEAK_SERVICE_TYPE_KEYBOARD service, using the command XPEAK_COMMAND_READ of the service XPEAK_SERVICE_TYPE_KEYBOARD. In this case, the value of the field ***PinEntryViaKeyboardService*** of the command XPEAK_COMMAND_GET_CAPABILITIES should be ***true***. Before the PIN validation, the application should execute the XPEAK_COMMAND_READ of the service XPEAK_SERVICE_TYPE_KEYBOARD with the parameter ***Mode*** with the value XPEAK_KEYBOARD_READ_MODE_PIN (0x7060104). In that moment, the user will type the PIN.
2	The PIN is inserted without the intervention of any XPEAK_SERVICE_TYPE_KEYBOARD service. In this case, the value of the field ***PinEntryViaKeyboardService*** of the command XPEAK_COMMAND_GET_CAPABILITIES should be ***false***. In this case only one command is necessary to validate a PIN. During the execution of this command, the user should type the PIN and the service will validate it.

Parameters:

- **`String Key`**
  Name of the key used to validate PIN. This key had to be imported with the XPEAK_SECURITY_USE_PIN (0x7060125) use. If the key has not XPEAK_SECURITY_USE_PIN (0x7060125) use, the error XPEAK_RESULT_INVALID_KEY (0x7060119) will be returned.

- **`String ValidationData`**
  Contains the data used to validate the PIN. The length of this data should be 8 bytes. Each byte is represented by two characters of the String value.

- **`String Offset`**
  Offset for the PIN block. If this parameter is the empty String (""), then no offset is used.

- **`int MaximumPinLength`**
  This parameter specifies the maximum number of digits of the PIN.

- **`String DecimalizationTable`**
  ASCII decimalization table compound by 16 characters included in the range ‘0’ to ‘9’. Used to convert the hexadecimal digits of the encrypted validation data to decimal digits.

Result:

- **`boolean IsValidPin`**
  If ***true***, the typed PIN is valid. If ***false*** the typed PIN is not valid.

- **`int Result`**

  Appart from the common values, this command could return as result:

Result	Description
XPEAK_RESULT_INVALID_KEY (0x7060119)	Error due to the key has not the properly uses for this operation.
XPEAK_RESULT_KEY_NOT_FOUND (0x706011A)	No keys with the specified name has been loaded in the security service.
XPEAK_RESULT_NO_TYPED_PIN (0x7060135)	The PIN has not been typed.
XPEAK_RESULT_WRONG_TYPED_PIN (0x7060136)	The length of the typed PIN is wrong or any of the pressed keys is not allowed.