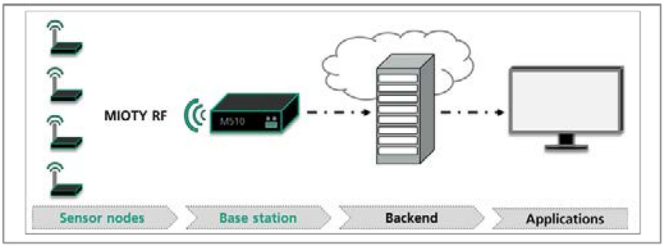
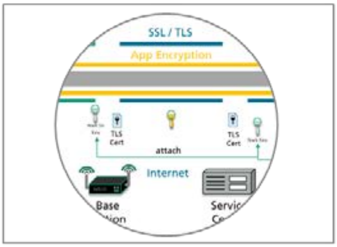
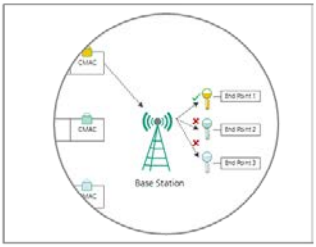
Fraunhofer IIS

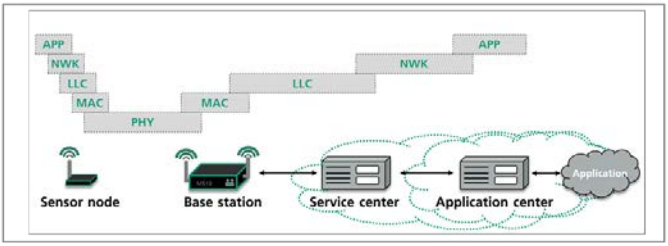
FRAUNHOFER INSTITUTE FOR INTEGRATED CIRCUITS IIS

MIOTY^{TM} – MAC AND HIGHER LAYERS









Published by
Fraunhofer Institute for Integrated Circuits IIS, Erlangen

Editorial team
Communication Systems Division: Dr. Gerd Kilian
Positioning and Networks Division: Josef Bernhard
Author: Dominik Soller, Nguyen Quynh Nhu Ho

Layout and production
Communication System Division: Marketing and Communication

Low Power Wide Area Networks (LPWAN) have emerged as a new wireless networking solution that aim to realize the Internet of Things (IoT) vision by addressing critical communication challenges of power efficiency, coverage, cost and scalability. Capitalizing on cutting-edge qualities of LPWAN with advanced enhancement on network performance, the MIOTY™ (My IoT) Technology delivers industry-grade connectivity that satisfies the most demanding (Industrial) IoT requirements.

MIOTY™ is a LPWAN solution dedicated for large-scale, private IoT Networks. Outstanding technical features and functionalities of the MIOTY™ system are empowered by its three core communication layers – the physical layer, the MAC (Media Access Control) layer and the LLC (Logical Link Control) layer. While the physical layer defines the unique MIOTY™ communication link with Fraunhofer IIS Telegram Splitting technology, the MAC and LLC layers are responsible for other network-specific tasks to complete the data transmission path. This document provides a detailed description of the MIOTY™ MAC- and LLC-layer specifications as well as the backend architecture.

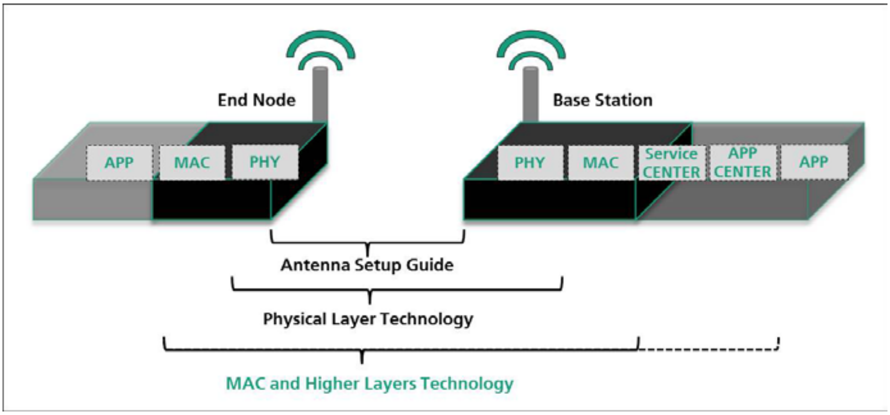


Table of Contents

List of abbreviations

BSSCI	Base Station Service Center Interface
CMAC	Cipher-based Message Authentication Code
CRC	Cyclic Redundancy Check
DL	Downlink
LLC	Logical Link Control
LPWAN	Low Power Wide Area Network
MAC	Medium Access Control
MPDU	MAC Protocol Data Unit
MPDUCNT	MPDU-Counter
MPF	MAC Payload Format
MSB	Most Significant Bit
NAT	Network Address Translation
RSSI	Received Signal Strength Indication
RX	Receiver
SCACI	Service Center Application Center Interface
TCP	Transmission Control Protocol
TLS	Transport Layer Security
UL	Uplink

1 Overview of the MIOTY™ system

1.1 Data chain in a MIOTY™ deployment

The MIOTY™ system architecture portrays a star network where one central base station aggregates low data-rate messages from thousands of battery-powered sensors operating at the edge. Enabled through the MIOTY™ RF Wireless Link characterized by the Telegram Splitting approach, communication between the base station and end nodes can be achieved over up to 15 km range (in flat terrain) utilizing worldwide license-free sub GHz bands (915 MHz – North America and 868 MHz – Europe).

MIOTY™ system provides two built-in security layers consisting of network-level and application-level encryption. The network-level encryption ensures that only authenticated devices and gateways can communicate with each other and messages cannot be accessed and interpreted by unauthorized entities during transmission. The application-level encryption enables end-to-end protection of user data, whereby encrypted sensor data can only be decrypted at the application center.

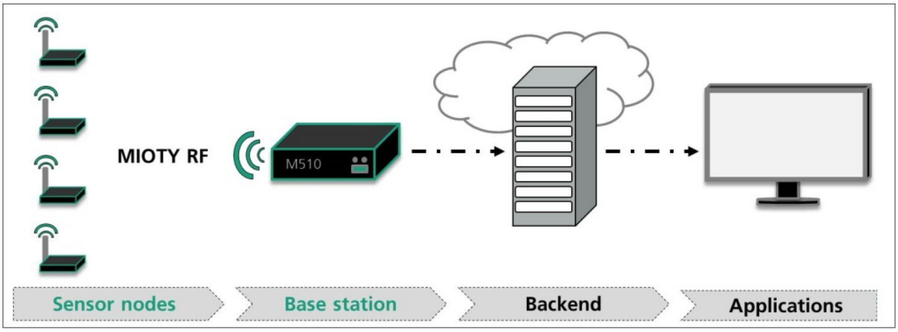


Figure 1: Data chain in a MIOTY™ deployment

Figure 1 provides an overview of the full data chain in a typical MIOTY™ deployment. Key components and their functions are further explained as follows:

  - **Remote sensor nodes** (end points) capture critical field data such as environmental parameters or machinery KPIs and transmit them to a base station at pre-defined time intervals using the MIOTY™ RF link.

- **The central base station** collects messages from all sensor nodes that have been attached to it[1], processes and decrypts network control data and forwards user data to backend through backhaul connection with higher throughput (e.g. Ethernet, Wi-Fi…). In bi-directional communication, it generates downlink[2] messages based on payload received from the backend. These messages are then transmitted to the respective sensor nodes for remote command and control applications. A MIOTY™ network can incorporate more than one base station, in which case, the same message can be received by multiple base stations.

- **The backend** entails a service center and an application center which resides on either a third-party cloud platform or an internal user server depending on customers' requirements.

  The service center is responsible for device and network management together with key management of the network-level cryptography. When two or more base stations receive and forward the same message, it additionally carries out de-duplication task by eliminating repeated data. In bi-directional communication, the service center further tracks downlink transmission slots and duty cycles, as well as decides which base station is allowed to answer which specific sensor nodes. All devices (sensor nodes and gateways) operating in the same MIOTY™ network are registered at the service center for administration and operational control.

[1] Attachment refers to the registration mechanism that authorizes a certain sensor node to communicate with a certain base station. Attachment can be done over-the-air via a base station or triggered by the server internally.

[2] Following common terminology in mobile radio system, uplink is the direction from the sensor node to the base station and downlink is the direction from the base station to the sensor node.

The application center serves as a point of contact with end users or application operators. Node credentials can be entered once here and accordingly transferred to the service center for registration of sensor nodes in the MIOTY™ network. When receiving messages from the service center, the application center decrypts and decodes user data by extracting application-specific information (e.g. temperature, pressure levels…) from the generic binary packets. Interpretable user data are then forwarded to the application platform.

- **Applications or application platforms** stand for any server or cloud computing systems whose core functions are data storage and analytics. Patterns can be identified and visualized on user interfaces for predictions and execution of timely responses.

1.2 Communication layers in the MIOTY™ system

The communication layer model depicts different function levels of a networking system. As illustrated in Figure 2, the full communication chain of a MIOTY™ network entails five main layers which are physical layer, MAC (Medium Access Control) layer, LLC (Logical Link Control) layer, network layer and application layer. These layers are handled by different components along the data chain. In particular, the physical, MAC and LLC layers represent the core of MIOTY™ software to be programmed on sensor nodes and gateways. Network and application layers, on the other hand, are specific to user applications and thus defined by the customers.

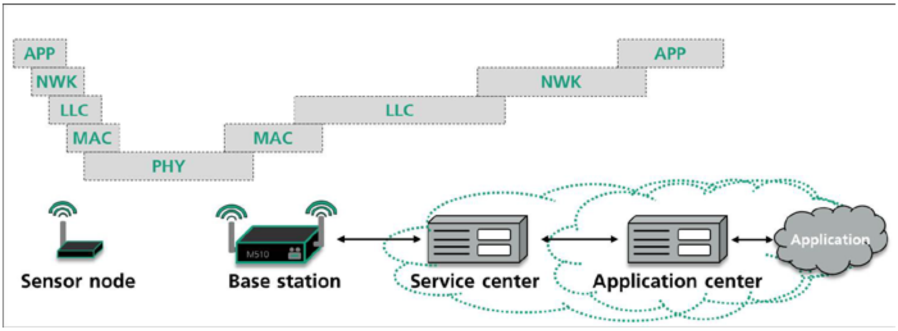


Figure 2: Communication layers in the MIOTY™ system

Based on Telegram Splitting technology – a unique transmission modulation developed and patented by Fraunhofer IIS, the **physical layer** characterizes the MIOTY™ RF Wireless Link and provides data connection between sensor nodes and the central base station. Besides enabling two core qualities of traditional LPWANs – high power efficiency and long transmission range, Telegram Splitting delivers other advanced technical features in the MIOTY™ system in terms of robustness, scalability and mobility (Details on the physical layer are provided in a separate "MIOTY™ – Physical Layer Technology" document).

Connecting the physical with higher layers in the communication stack, the **MAC layer** provides channel access so that sensor nodes and the central base station can communicate with each other through the MIOTY™ RF. Key functions of the MAC layer include address resolution, validation of message integrity and authenticity, decryption of network-level security, and timing control of downlink messages.

- <u>Address resolution, authentication and message validation</u>: every MIOTY™ sensor node is assigned with a lifetime IEEE standardized EUI64 address at production for its unique identification. This address information is spread into address-hint and CMAC (Cipher-based Message Authentication Code) tags within a data packet. When a message is received at the base station, the MAC layer is responsible for deriving the complete EUI64 address information from these tags to identify from which sensor node, the respective message has been sent. Additionally, by decoding the CMAC tag, the MAC layer controls whether the authenticity and integrity of a data packet remains intact after the transmission.

- Network-level cryptography: each communication link between the sensor node and the base station is encrypted with a session-specific network key using AES 128 mechanism to protect control and user data during transmission. At the base station, the MAC layer is accountable for decrypting this network-level security so that data can then be interpreted and processed at higher layers.

- Timing control of downlink messages: in bi-directional communication, a downlink message is sent from the base station approximately 6 seconds after the reception of an uplink message. Storing entry timestamps of all uplink messages and controlling the downlink window, the MAC layer ensures that downlink messages are issued to the corresponding end nodes at the right time.

Residing above the MAC layer, the **LLC layer** handles network management information (control data) such as attachment and detachment command, link adaptation command or downlink query by the base station for reception parameters. While physical and MAC layers are operated exclusively at the base station, the LLC layer is partially administered by the service center.

Network and application layers go beyond the scope of the MIOTY™ communication system, which is designated only for data transfer between sensor nodes and central base station. These two layers, therefore, are mostly defined by the users. The network layer basically undertakes the task of transporting user data to the application center, where the information is decoded and processed. The application layer is responsible for data presentation and other application-specific services.

2 MAC layer

The MAC layer controls the timing of transmissions and keeps track of transmission windows. Furthermore it is responsible for address resolution and provides network level cryptography and authentication.

2.1 Transmission timing

In a MIOTY™ network uplink transmissions can be initiated by the end points at any time. Downlink transmissions though are only possible after a preceding uplink transmission as illustrated in Figure 3. This is due to the fact that end points are usually battery powered and designed for runtimes of multiple years. Thus end points remain in a low power state for most of the time and cannot receive incoming downlink transmissions. Only after signaling an opened downlink window in an uplink transmission the end point will be actively receiving at the specific time offset of 16384 symbol durations (approximately 6.88 seconds) from the end of the uplink transmission. More details about the transmission timing can be found in the "MIOTY™ – Physical Layer Technology" guide.

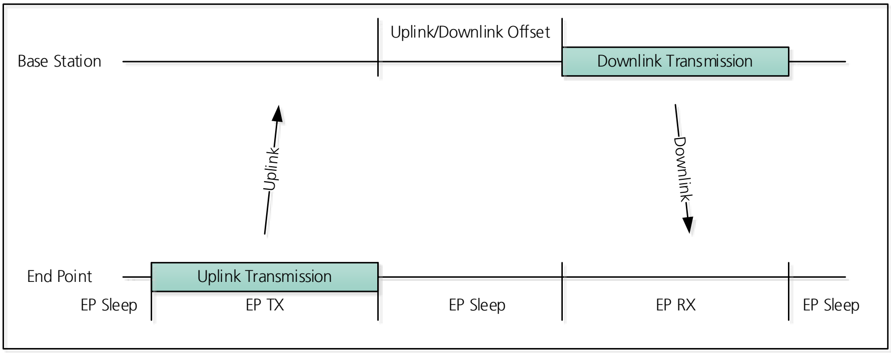


Figure 3: Uplink/downlink transmission scheduling

The MAC needs to keep track of any opened downlink windows for every end point and provides the precise timing information from the according uplink transmission back to the physical layer in case of a downlink transmission to ensure correct alignment with the end point reception window. To mitigate the strict timing constraints for downlink transmissions, it is also possible to preschedule data for future downlink windows of an end point in the base station.

2.2 Network level cryptography

MIOTY™ requires the encryption of every message payload with the exception of over-the-air attachment requests. This network level cryptography is handled by the MAC layer. It ensures that no user data is transmitted in plaintext even when no application level encryption is used and also protects link layer control information. The network level cryptography is based on an end point specific network key and the industry standard AES128 cipher algorithm [1].

As secure over-the-air key exchange procedures are not feasible with strict bandwidth and processing power limitation, every end point is assigned a preshared network key at production. This preshared key is the basis for all encryption purposes during the end points life time and must be kept secret. A separate preshared key should be used for the strongly recommended application level encryption. This second key can be kept private to the application and is not required by the MIOTY™ system. Please refer to the "MIOTY™ – Security Guideline" document for more details about the security concept in general.

For bi-directional end points the network key is session specific, it is then derived from the preshared network key during the attachment procedure. Unidirectional end points cannot initiate over the air attachments and thus operate in one lifetime session utilizing the preshared network key. More details about the attachment and the network key generation can be found in section 3.1.

For an LPWAN technology like MIOTY™ it is important to avoid payload overhad induced by padding to satisfy fixed cryptographic block sizes. Furthermore decryption of a message cannot depend on previous transmissions as there is no guaranty for successful reception, especially in the case of unidirectional communication. To support arbitrary payload sizes without padding while allowing independent processing of every transmission, the AES128 algorithm in counter mode of operation [2] is employed.

In counter mode, depicted in Figure 4, an initialization vector or nonce is used as input for a block cipher to create a secret seed vector. A counter contained in the initialization vector must be incremented for every processed block to ensure no seed vector is ever reused for a second block. The seed vector is then combined with the plaintext of the message via an exclusive or operation to form the ciphertext. As the exclusive or operation maps the data bit by bit, the ciphertext can be truncated to the size of the plaintext for shorter plaintexts. If the plaintext exceeds the size of one block, the counter in the initialization vector is incremented and the procedure is repeated until the entire plaintext message is encrypted.

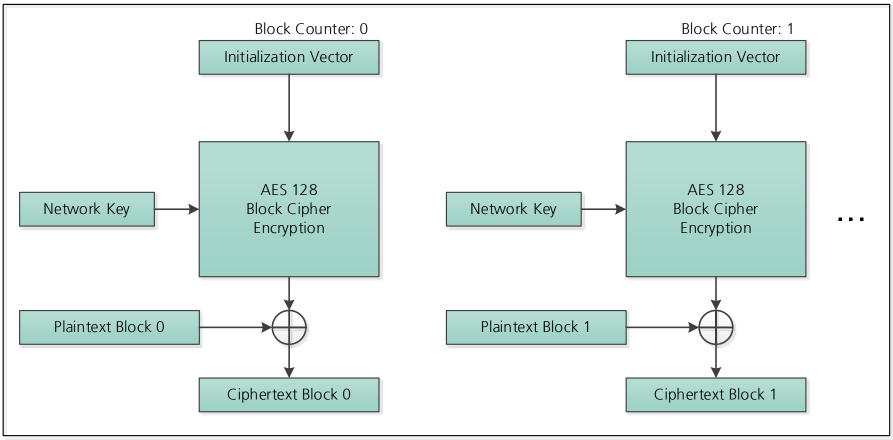


Figure 4: Encryption via block cipher in counter mode of operation

The exclusive or operation is trivially reversible thus reusing the same initialization vector for a second block compromises security. Attackers might then recover the seed vector via known plaintext or gain information from bitwise comparison of different blocks. MIOTY™ therefore includes a block counter and a packet counter in the initialization vector, depicted in Figure 5.

The 16 bit block counter is incremented for every block within one packet and is reset to zero at the beginning of a packet. The 32 bit packet counter is incremented for each packet and can only be reset via an over-the-air attachment procedure. Thus for unidirectional end points the packet counter must never be reset, limiting the lifetime transmissions to approximately 4 billion messages. This corresponds to approximately 540 years of operation at continuous transmission of one message every 4 seconds.

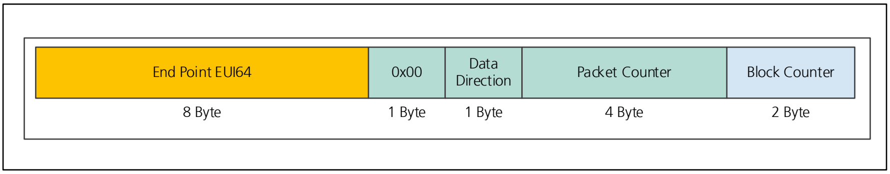


Figure 5: Initialization vector for counter mode

Additionally the initialization vector includes the end point EUI64 and a data direction indicator. The data direction indicator is set to 0x00 for uplink transmissions and 0x01 for downlink transmissions. This ensures different initialization vectors for uplink and downlink direction even when the same counter is used. This is important to avoid any dependency of an unused counter position on transmissions by the other side which might be lost or corrupted. The inclusion of the EUI64 furthermore provides a unique sequence of initialization vectors if the same network key is used for multiple end points. It is still strongly recommended to use individual keys for every end point for a maximum of security.

Decryption is identical to encryption in counter mode of operation with plaintext and ciphertext exchanged. The block cipher is again used for encryption of the initialization vector but then the ciphertext is used as second input for the exclusive or operation. The used initialization vector must be known to the recipient. Therefore the end point identity and the packet counter needs to be extractable from unencrypted header information included in the transmission. Details about the extraction of this information are discussed in the following sections.

2.3 Message authentication

Every MIOTY™ message is authenticated on the MAC layer with a Cipher-based Message Authentication Code (CMAC) [3] [4]. A CMAC is based on a cryptographic key and a block cipher to generate an individual bit sequence verifying the message data. In MIOTY™ the network key of the end point and the AES128 cipher are employed for the CMAC computation. The CMAC also supplements the CRC (Cyclic Redundancy Check) in the physical layer as a secondary integrity check.

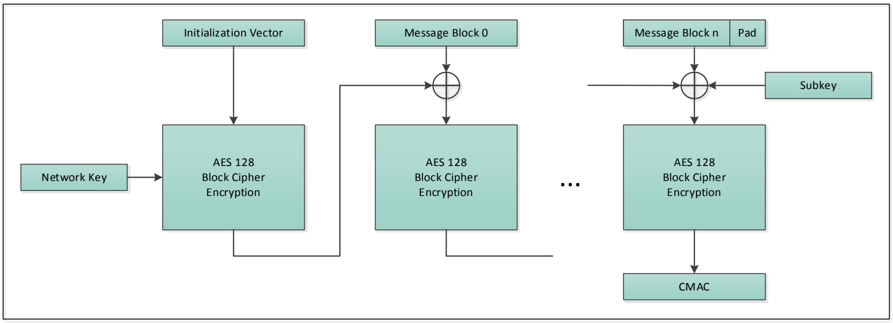

Initialization Vector
Message Block 0
Message Block n
Pad
Subkey
Network Key
AES 128 Block Cipher Encryption
AES 128 Block Cipher Encryption
...
AES 128 Block Cipher Encryption
CMAC

Figure 6: CMAC computation procedure

As illustrated in Figure 6, the message, preceded with an initialization vector, is injected into the CMAC algorithm block by block until the entire message has been processed. Additionally a subkey K1 is introduced with the last block. If the message size does not adhere to block boundaries, the last block is padded and processed with a second subkey K2.

The two subkeys are derived from the main key, which is the network key in MIOTY™, as shown in Figure 7. First a vector of all zeros is encrypted with AES128 using the network key. If the most significant bit (MSB) of the cipher-text is zero, the subkey K1 is the ciphertext shifted to the left by one bit. Otherwise the ciphertext is additionally combined with a special constant via an exclusive or operation after shifting it to the left by one bit to form subkey K1. Subkey K2 is then derived from subkey K1 by again either a sole left shift or a left shift followed by an exclusive or operation with the special constant depending on the MSB value. The reason for this procedure is beyond the scope of this document; please refer to the CMAC specification [3] for more information.

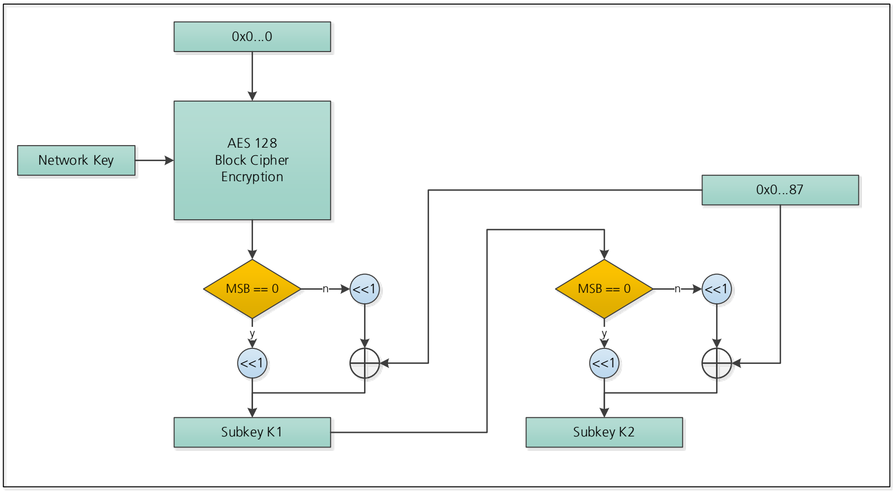


Figure 7: CMAC subkey generation procedure

The initialization vector used in MIOTY™ for the CMAC computation corresponds to the initialization vector of the network encryption with the block counter set to 0xFFFF, displayed in Figure 8. Setting the block counter to a value which can never be used for encryption of a message block avoids leaking any information about a valid block initialization vector via the CMAC.

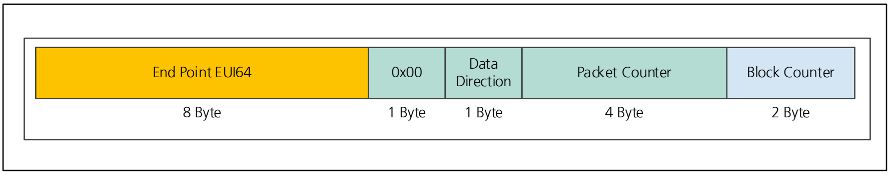


Figure 8: Initialization vector for CMAC computation

The insertion of the initialization vector ensures, that all information required for a valid deciphering of the message is also signed via the CMAC. To minimize overhead MIOTY™ does not include all information explicitly in a transmitted packet but instead relies on recovery of implicit information. Thus merely signing the message contents is not sufficient to guarantee the used parameters on end point and base station side match. In the case of invalid information garbled data might then be propagated as a valid message.

To verify a CMAC, the recipient simply computes the expected CMAC for the message and compares the result of this computation to the received sequence. This also allows truncation of the CMAC to an arbitrary width to accommodate different requirements for false positive resilience or bandwidth limitations.

2.4 Address resolution

Every MIOTY™ end point is assigned a lifetime IEEE standardized EUI64 [5] from the vendor's identifier pool at production. Optionally the EUI64 can be preceded with a 64bit IPv6 subnet to form a full 128bit IPv6 address. In this case the entire IPv6 address is considered as permanently assigned to the end point and cannot be reassigned until the end point is retired. The IPv6 subnet is intended to enable future IPv6 mobility based enhancements.

To reduce overhead the full eight byte EUI64 address of the end point is not required in every MIOTY™ transmission. Instead all attached end points are assigned a network local 2 byte short address. Short addresses are not unique and can be shared by many end points. The assignment of the short address can either be done during an over the air attachment procedure or it can be preassigned. For unidirectional end points a preassignment is required. Details about the attachment procedures are discussed at the LLC layer in section 3.1.

The usage of non-unique short addresses not only reduces the data overhead in the MAC layer but also obfuscates the identity of the end points against third parties. Of course it must be noted, that with sufficient effort the identity can still be revealed i.e. via triangulation of the radio transmissions. Especially in smaller network setups a short address might also be assigned to one end point only.

The resolution of the end point EUI64 is done in the MAC layer by combining the explicit addressing information provided by the short address with implicit information about an end points identity contained in the CMAC signature of the packet. In other words, the employed cryptographic key as well as the inclusion of the EUI via the initialization vector ties a CMAC to one specific end point. The recipient can compute a CMAC for the packet under the assumption, that the packet is associated with a certain end point. If the CMAC matches the assumption is considered valid. Figure 9 illustrates this address resolution procedure.

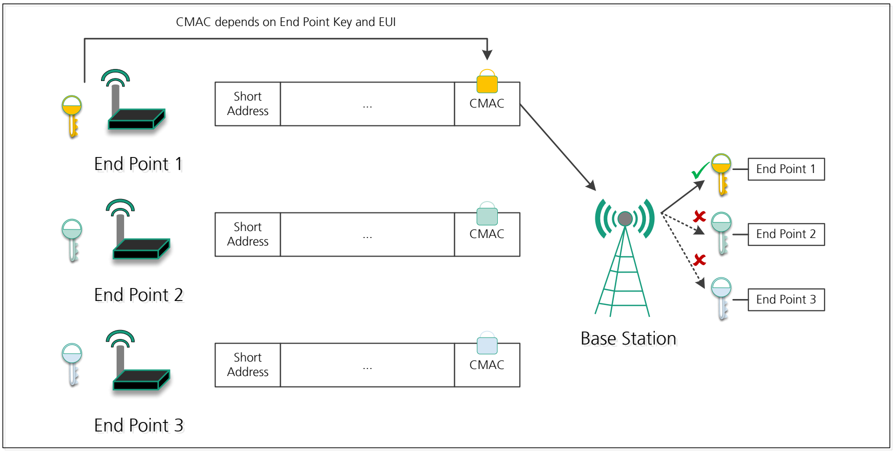


Figure 9: Address resolution via short address and signature

The short address is merely required in uplink direction to narrow down the range of eligible end points for the base station to probe against the packet signature. In downlink direction the short address is entirely omitted as the timing window does provide a preselection already and only one signature check is required on the end point side anyway. The short address can thus also be seen as an address hint for the base station.

Without this constraint to a subset of end points, address resolution would be infeasible for larger network setups with many thousands or even millions of nodes. Not only because of the excessive processing effort required to compute every alleged signature for every end point in the base station, but also because of the limited entropy of the signature.

Extracting implicit addressing information from the signature cannot provide any gain in transmitted information. The total amount of information comprised by the transmission is still limited by the number of bits transmitted. Thus the signature aided address resolution must be paid for with a reduction in resilience of the authentication. If multiple end points are in the subset selected by the short address, multiple bit sequences form a valid signature for this message, one for each end point. With the 16 bit short address in MIOTY™ the average subset in a network with one million end points is 16. Thus the signature is effectively reduced by 4 bits considering the invalid packet rejection.

The extraction of implicit information from the signature still yields some benefits over the usage of a longer, unique address with a shorter signature. So the protection against forgery of a transmission for a specific end point remains uncompromised as still only one valid signature is associated with one end point. Considering the additional integrity check on the physical layer, the most likely cause for a false positive is a dedicated attack. Such an attack becomes less effective if no specific end point can be targeted.

Furthermore forfeiting uniqueness for the short address also removes the requirement of a coordinated address assignment. This is especially important for unidirectional end points with preassigned short addresses. So while a 20 bit short address would suffice to uniquely identify one million end points, collisions almost certainly occur for an uncoordinated preassignment. With the signature assisted approach in MIOTY™ the global uniqueness of the EUI contained in the signature is utilized to mitigate collisions.

Ambiguities can still occur when mapping a signature to an end point though the probability is extremely low for practical network setups. Specifically for every end point the probability for a collision with any other end point sharing the same short address is one out of 2^{32}. For a typical scenario with 16 end points per address hint this means one ambiguity every 268 million packets on average. In this case the packet is discarded by the MAC layer to avoid propagation of potentially mismatched data.

Another advantage of the MIOTY™ concept is the avoidance of a hard boundary for the total number of end points like a 20 bit unique short address had. Instead the address space is dynamically adapted to the size of the network, shifting entropy from the authentication towards addressing.

2.5 MAC formats in uplink

The MAC layer format is based on the MAC protocol data unit (MPDU). The MPDU is transferred via the physical layer as payload and then provided back to the MAC layer on the base station. Figure 10 depicts the MPDU for packets in uplink direction.

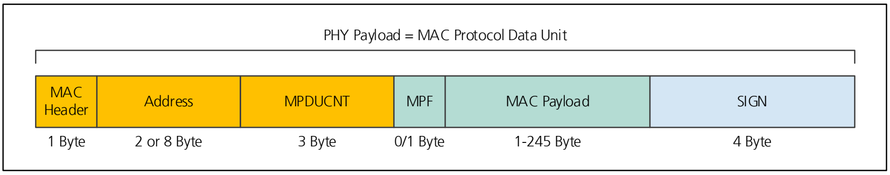


Figure 10: MAC protocol data unit for uplink transmissions

Byte order for all fields is most significant bit first. Details about the fields are discussed in the following sections.

2.5.1 Header

The MAC header is a one byte wide bit field used for signaling of various MAC options. The flags with the according options are listed in Table 1.

Bit	Function	Description
0	MAC version	0: Initial Version 1: Reserved
1	MPF flag	0: No MPF Field present 1: MPF Field present
2	Control flag	0: No control payload present (data only packet) 1: Control payload present
3	Response flag	0: No response expected

		1: Response expected
4	RX open flag	0: No receive window for DL reception opened 1: Receive window for DL reception opened
5	Addressing mode	0: Short addressing mode, 16 bit short address 1: Full addressing mode, 64 bit EUI
6	Attach flag	0: Regular packet 1: Attachment packet, also requires addressing mode = 1, control flag = 1, RX open flag = 1 and response flag = 1
7	ACK	0: No downlink packet received since last uplink transmission 1: Downlink packet received

Table 1: MAC Header for uplink transmissions

MAC version

The initial MAC version is indicated with a 0, 1 is reserved for future extensions.

MPF flag

The MPF flag signals if a MAC payload format field is included in the MPDU. The MPF field can be used to provide information about the payload format with every packet. This information is then propagated to the higher layers responsible for the payload processing.

Control flag

The control flag signals the presence of control information within the payload. This information is provided to the LLC layer which handles further processing of the control information. The control flag is not strictly MAC related however its inclusion in the MAC header removes the necessity for a separate LLC header.

Response flag

The response flag indicates the end point´s expectation of a response from the base station. If the base station has no pending downlink data it might just respond with an acknowledgement. A pure acknowledgement of an uplink transmission does not require any MAC payload but is handled via an authenticated sync sequence in the physical layer downlink. Please refer to the "MIOTY™ – Physical Layer Technology" document for more information.

RX open flag

The RX open flag signals the availability of a downlink window at the end point without requesting a response. The base station can utilize this downlink window for a downlink transmission or discard it if no pending data is available. For power consumption reasons an end point might not offer a downlink window after every uplink transmission. If the response flag is set, the RX open flag needs also to be set.

Addressing mode

The addressing mode switch selects between short addressing and full addressing modes. Accordingly the address field contains the 16 bit short address or the 64 bit EUI of the end point.

Attach flag

The attach flag is used only for over-the-air attachment requests. These requests are unencrypted and cannot yet be assigned to an attached end point and therefore need to be processed differently in the MAC layer. Attachment packets always include control information and require a response thus the according flags must also be set. More details about the attachment procedure are provided at the LLC layer section 3.1.

Acknowledge flag

The acknowledge flag signals the successful reception of a downlink transmission after the last uplink transmission. As downlink transmissions can only be sent after an uplink transmission and are thus directly linked to an uplink transmission, only one downlink transmission can occur between uplink transmissions.

2.5.2 Address

Depending on the addressing mode flag in the MAC header, the address field contains either the 16 bit short address or the 64 bit EUI of the end point. In short addressing mode the end point identity is determined via signature aided address resolution as described in section 2.4.

2.5.3 Counter

The MPDUCNT (MPDU-Counter) field is filled with the 24 least significant bits of the 32 bit packet counter shown in Figure 11. The eight most significant bits are not explicitly transmitted and must be tracked over previous transmissions in the base station. As stated in section 2.2 the same packet counter must never be reused for another uplink transmission while the same network key is utilized. Thus the packet counter must be incremented after each transmission and is only reset by an over-the-air attachment. This also protects against replay attacks as previous counter values are rejected.

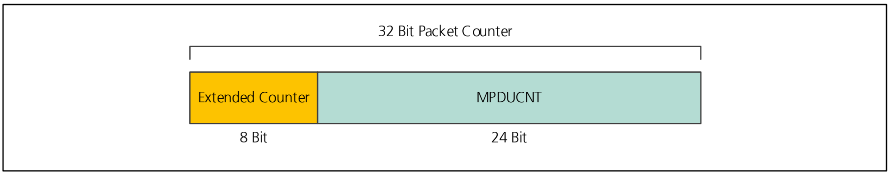


Figure 11: Aggregated packet counter

For networks with less frequent transmissions or a limited operation period, the tracking of the extended counter might be omitted. Unidirectional end points are then limited to approximately 16 million lifetime messages, bidirectional end points are required to reattach after exhausting the 24 least significant bits of the packet counter.

Attachment request packets contain a separate 24 bit attachment counter value in the MPDU field instead of the packet counter. It is important to note, that MAC layer procedures still rely on the packet counter, the packet counter therefor is implicitly assumed 0 for attachment request packets. Actually the attachment counter is link layer information and is only transmitted in the MAC layer counter field to reduce overhead. The attachment counter is incremented after each attachment request packet. More details about the attachment procedure are provided in the LLC layer section 3.1.

2.5.4 Payload format indicator

The MPF field is only present if the MPF flag in the MAC header is set to 1. It provides information for the interpretation of the following payload data. For most applications the payload format should be associated with the end point and therefore does not need to be transmitted via the MIOTY™ radio link. The specified values for the MPF field are listed in Table 2.

MPF Field	MPF Name	Description
0x00 – 0xBF	Reserved	Reserved
0xC0 – 0xFF	Custom	User specific payload format

Table 2: MAC payload formats

The MAC layer merely extracts the payload format information and propagates it along with the payload to the higher layers. No format specific interpretation of the payload is done within the MAC layer. For the MAC layer the MPF field can be considered as part of the payload which is separated as a service for the layers above to avoid dedicated signaling of the presence within the payload. The MPF field is also covered by the network layer encryption.

2.5.5 Payload and encryption

The MAC payload field contains the ciphertext of the variable size data exchanged with the link layer. The actual size is determined as the remainder between the total MPDU size, provided by the physical layer, and the size of the other MPDU fields. A maximum of 245 bytes is available if short addressing mode without MPF is used. Full addressing mode or the presence of an MPF field reduces the maximum payload size accordingly to adhere to the physical layer limit of 255 bytes for the entire MPDU.

The MAC payload field and, if present, the MPF field are encrypted with the network layer cryptography according to section 2.2 for every standard packet. For attachment request packets no cryptography is applied as no network key is available to the base station yet.

2.5.6 Signature

The signature field contains a CMAC based signature which cryptographically signs the entire preceding MPDU contents. The computation of the CMAC is done according to the procedure described in section 2.3. Figure 12 shows the input data vector for the CMAC algorithm including the initialization vector. The SIGN field is filled with the 32 most significant bits of the resulting CMAC.

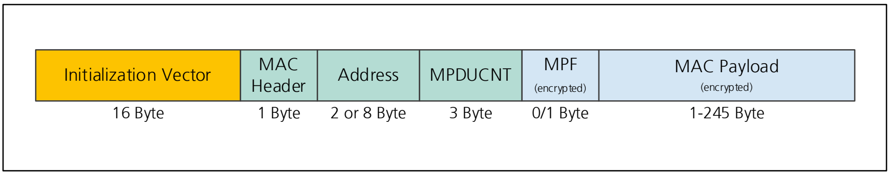


Figure 12: Input data for uplink signature computation

The optional MPF field and the payload are injected into the signature in encrypted form (encrypt-then-MAC). This avoids any theoretical leakage of information about the plaintext through the signature and also allows signature checks without prior decryption. This is especially advantageous considering the signature aided address resolution used by MIOTY™.

In the special case of an attachment request packet (attach flag set) where no network key is available yet, a key of all zeros is used instead for the CMAC computation. The function of the signature is then reduced to verification of the packet integrity. The authenticity of an attachment request is verified within the attachment procedure described in the LLC section 3.1. Please also note the implicit packet counter of zero used within the initialization vector in this case as mentioned in section 2.5.3.

2.6 MAC formats in downlink

The MAC layer format in downlink direction is based on the downlink MPDU depicted in Figure 13. In contrast to the uplink MPDU the address and the MPDUCNT fields are omitted. The addressing information is implicitly provided by the timing of the transmission according to the end point downlink window. The signature check rejects other transmissions accidentally aligning with the downlink reception window. As every downlink transmission is tied to a preceding uplink, the uplink packet counter is also associated with the downlink.

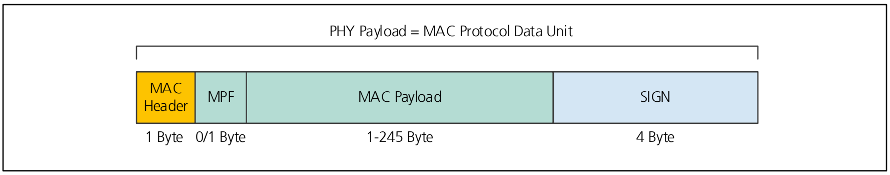


Figure 13: MAC protocol data unit for downlink transmissions

The MPDU and the total size of the MPDU are provided to the physical layer. Byte order for all fields is most significant bit first for all MAC fields. Details about the fields are discussed in the following sections.

2.6.1 Header

The MAC header is a one byte wide bit field used for signaling of various MAC options. The flags with the according options are listed in Table 3.

Bit	Function	Description
0	MAC version	0: Initial Version 1: Reserved
1	MPF flag	0: No MPF Field present 1: MPF Field present
2	Control flag	0: No control payload present (data only packet) 1: Control payload present
3	Response flag	0: No response expected 1: Response expected
4	RX open flag	0: No further downlink window required 1: Request downlink window after next uplink
5	Response priority flag	0: Response expected within regular time window 1: Response expected within priority time window
6/7	Reserved	Reserved, must be set to 0

Table 3: MAC Header for downlink transmissions

MAC version

The initial MAC version is indicated with a 0, 1 is reserved for future extensions.

MPF flag

The MPF flag signals if a MAC payload format field is included in the MPDU. The MPF can be used to provide information about the payload format with every packet. This information is then propagated to the higher layers responsible for the payload processing.

Control flag

The control flag signals the presence of control information within the payload. This information is provided to the LLC layer which handles further processing of the control information. The control flag is not strictly MAC related however its inclusion in the MAC header removes the necessity for a separate LLC header.

Response flag

The response flag can be set by the base station to indicate the expectation of a response from the end point. This might either be to acknowledge the reception of the downlink or to open further downlink windows in the case of additional pending downlink data. The RX open flag and response priority flags might be used in combination with the response flag to specify the request.

RX open flag

The RX open flag signals the request from the base station for an additional end point downlink reception window after the next uplink transmission from the end point. This flag might also be used in combination with the response and response priority flag to control the time frame within which the additional downlink window should be offered by the end point.

Response priority flag

The response priority flag can be used by the base station to indicate the urgency of the response by the end point. If no priority response is requested, the end point might respond according to its regular transmission schedule while a priority response should occur as soon as possible. The specific behavior is end point dependent and cannot be enforced by the MAC layer.

2.6.2 Payload format indicator

The MPF field is only present if the MPF flag in the MAC header is set to 1. More information can be found in the uplink MPF field description in section 2.5.4.

2.6.3 Payload and encryption

Like in uplink direction, the MAC payload field contains the ciphertext of the variable size data exchanged with the link layer. The maximum size increases to 250 bytes due to reduced MAC overhead in downlink direction in combination with the identical 255 byte limit for the aggregated MPDU.

The MAC payload field and, if present, the MPF field are encrypted with the network layer cryptography according to section 2.2.

2.6.4 Signature

The signature field, analogous to the uplink signature, contains a CMAC based signature which cryptographically signs the entire preceding MPDU contents. The computation of the CMAC is done according to the procedure described in section 2.3. Figure 14 shows the input data vector for the CMAC algorithm including the initialization vector. The SIGN field is filled with the 32 most significant bits of the resulting CMAC.

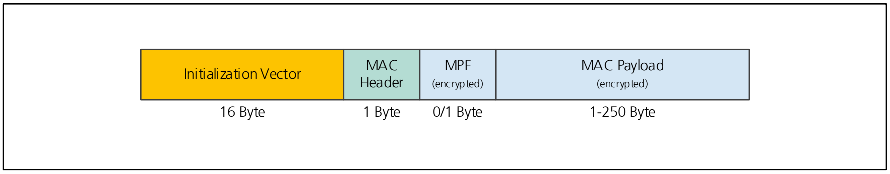


Figure 14: Input data for downlink signature computation

A separate signature is used by the downlink physical layer as an authenticated synchronization sequence. This signature is also provided by the MAC layer alongside the MPDU. It does not authenticate the downlink transmission it is used for, but instead authenticates the reception of the according uplink transmission from the end point. Also it must be known to the end point before reception to allow usage as a synchronization sequence. Therefore the synchronization signature is generated with only the initialization vector as input. The information in the initialization vector depends merely on the end point EUI64 and the packet counter and thus is known after the uplink transmission.

3 LLC layer

The logical link control layer or simply link layer handles control procedures between MIOTY™ end points, base stations and the service center. Most importantly this includes the attachment and detachment of end points and the adaptation of radio link parameters.

3.1 Attachment and detachment

Before an end point can be serviced by a base station, the end point must be attached to the base station. The end point attachment procedure provides the base station with essential information about the end point and associates the end point with a short address. Bi-directional end points may initiate an over-the air attachment procedure while unidirectional end points are pre-attached at production. In either case the attachment information can be propagated via the backend to multiple base stations.

3.1.1 Attachment via the service center

In an attachment via the service center the base station receives all information required to service an end point from the service center. The end point is not directly involved in this procedure and might not even be within the coverage of the base station. For unidirectional end points this is the only option for an attachment but it is also used to propagate attachment information received via an over-the-air attachment to other base station as will be explained in the next section.

As depicted in Figure 15, the procedure is started by the service center with an attach propagation message to the base station. It contains the network key for the end point, the short address associated with the end point as well as the radio configuration of the end point. These attributes can either be the factory provisioned attributes of a unidirectional end point or the result of a preceding over-the-air attachment. The base station then accepts the attachment. Any future transmissions of the end point can be processed in the base station based on the acquired information.

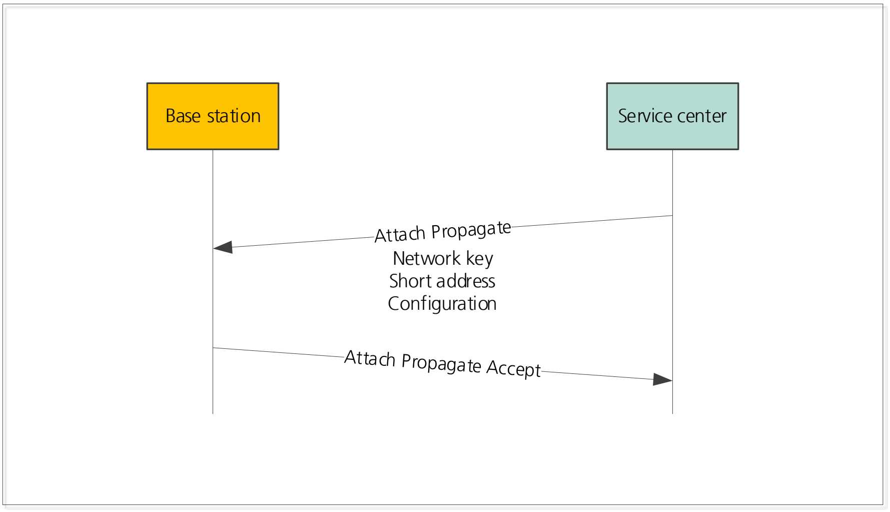


Figure 15: Service center based attachment procedure

3.1.2 Attachment over the air

Bi-directional end points have the option to initiate an over-the-air attachment procedure, shown in Figure 16, to gain access to a MIOTY™ network. The end point therefor sends a special attachment packet, which can be received by any compatible base station within coverage. This attachment packet is not encrypted and uses full addressing mode to allow processing in the base station MAC layer without further information about the end point. Details about the MAC formats for attachment packets can be found in section 2.5.

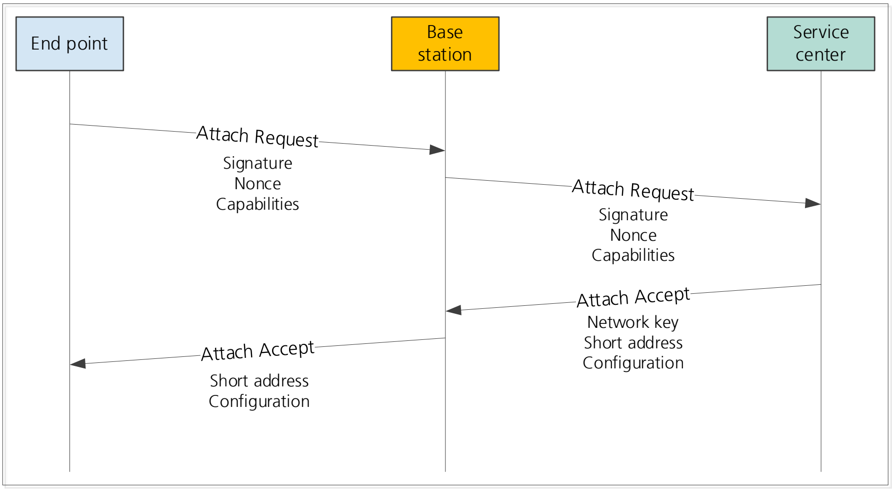


Figure 16: Over-the-air attachment procedure

The attachment request packet contains information about the end point radio capabilities as well as a signature and a nonce. The signature allows verification of the end point identity in the backend before granting the attachment. The nonce is intended for generating a session specific network key based on the preshared network key in the backend.

After receiving the attachment packet from the end point, the base station forwards the requests to the service center. If the end point is known in the service center and the authenticity of the attachment request can be verified via the signature, the session specific network key is generated based on the nonce. The session specific network key is used for all network level cryptographic purposes, thus only the session specific key is required by the base station, the preshared key remains in the backend.

The end point is also assigned a short address for future transmissions. Depending on the network architecture, short address assignments might either be managed by the individual base stations or by the service center. The same applies for the selection of a radio configuration based on the advertised capabilities of the end point.

Finally the base station responds to the end point with an attachment accept downlink transmission containing the assigned short address and the radio configuration based on the end points capabilities. It is sent within the downlink window opened by the uplink attachment packet and uses a regular, encrypted packet based on the newly generated session specific network key. The end point can generate the same key locally based on the nonce sent in the attachment request, allowing normal processing of the downlink response by the end point. The response based on the valid session specific network key also legitimates the base station for the end point as unauthorized base stations have no access to this key.

In a network with multiple base stations the over-the-air attachment procedure is always handled by one base station exclusively. If the request is received by more than one base station, the service center selects one of these base stations to complete the attachment. When the over-the-air attachment is completed, the service center might propagate the negotiated information to other base stations in the network. This propagation resembles an attachment via the service center described in 3.1.1. The end point can then operate at all base stations in the network without reattaching.

3.2 LLC formats

As the link layer procedures not only involve data exchange over the MIOTY™ radio link but also hook into the backend, specifically the service center, the link layer actually has two interfaces. This section focusses on the link layer formats for the MIOTY™ radio communication between end point and base station. The link layer formats used between base station and service center are part of the BSSCI (Base Station Service Center Interface) described in section 4.2.

The foundation for the LLC layer is the LLC Protocol Data Unit (LPDU) which corresponds to the MAC payload as shown in Figure 17. It is exchanged with the MAC layer alongside some meta information. This meta information includes the total size of the LPDU as well as some LLC flags included in the

MAC header. Therefore the LPDU does not require a separate header field and a zero overhead is achieved for pure data packets.

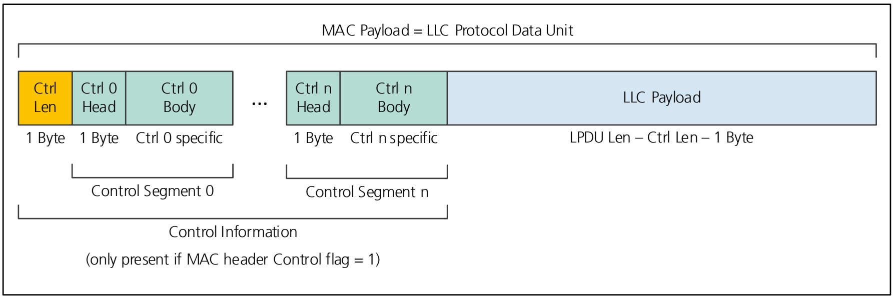


Figure 17: LLC protocol data unit

Link layer procedures are handled via optional control segments inserted at the beginning of the LPDU. The presence of control information is signaled via the control flag in the MAC header (see sections 2.5.1 and 2.6.1). Only when control information is included in the packet, the first byte provides the total length of all control segments.

The usable payload length is reduced by the size of the control information to adhere to MAC and physical layer packet size restrictions. Therefore the link layer must consider the user provided payload when adding control segments to a packet. If the packet size is insufficient to carry all control segments, the control information must be spread across additional packets.

3.2.1 Control segments

Control segments are comprised out of a header and an optional body. The segment header defines the type of the control segment. Every control segment type is associated with a fixed length for the control body, or no body at all for flag like control segments. Thus control segments are evaluated sequentially to determine the beginning of the next segment from the previous segment position and size.

In the case of future extensions, newer protocol version control segments can be attached behind previous version control segments. This allows a recipient with an older protocol version to process the supported segments and skip any remaining, unsupported segments via the total control length at the beginning of the control information.

An overview of the available control segments is provided in Table 4. User specific control segments cannot be processed by the standard LLC layer and require a customized implementation.

Control Header	Direction	Control Segment Name	Control Segment Size
0x10		Attach Request	11 Byte
0x11	UL	Attach Request IPv6	19 Byte
0x14	DL	Attach Accept	5 Byte
0x18	UL,DL	Detach Request	5 Byte
0x1C	DL,UL	Detach Accept	5 Byte
0x20	DL	DLRX-Status Query	1 Byte
0x21	UL	DLRX-Status Response	3 Byte
0x24	DL	Link Adaptation Request	3 Byte
0x25	UL	Link Adaptation Confirm	1 Byte
0xC0-0xFF	UL,DL	User specific control segments	User specific
Others		Reserved	

Table 4: LLC control segments

With the exception of the downlink RX status response, these control segments apply to bi-directional end points exclusively. Unidirectional end points have very limited link layer functions and rely on the backend to handle link layer procedures.

Attach request

The attach request control segment is used exclusively by end points within attachment packets to initiate an over-the-air attachment. As illustrated in Figure 18, it contains an end point info field, a nonce and a signature. The end point info field carries information about the end points radio capabilities; please see section 3.2.2 for details. Nonce and signature are propagated to the backend. From the link layer perspective these are entirely user specific and are not processed in any way. The intended purpose of those fields is discussed in section 3.1.2.

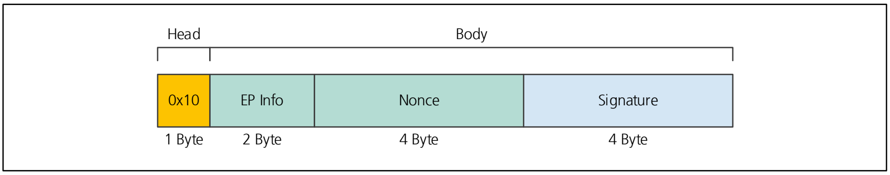


Figure 18: Attach request control segment

An alternate attach request segment, shown in Figure 19, is available for IPv6 enabled end points. It is extended with a field for the IPv6 subnet the end point is associated with. A full IPv6 address can be formed in combination with the end point EUI.

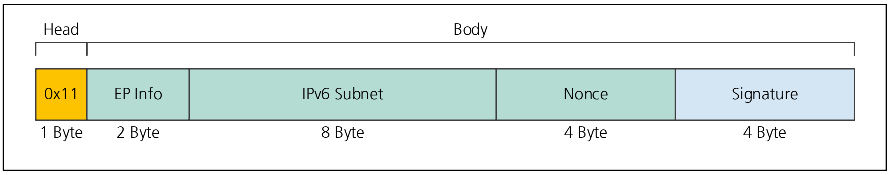


Figure 19: Attach request control segment for IPv6 enabled end points

Attach accept

The attach accept control segment is used by the base station as a response to an over-the-air attachment request. It selects the actual radio settings for future transmissions from the advertised end point capabilities and assigns a short address to the end point. Figure 20 depicts the control segment with the according end point info and short address fields.

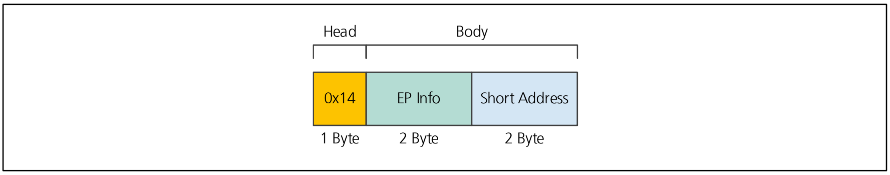


Figure 20: Attach accept control segment

Please note, that there is no requirement for a signature field in the attach accept control segment as the possession of the valid network key already verifies the legitimation of the base station by the backend to the end point. More details can be found in section 3.1.2.

Detach request

The detach request control segment, shown in Figure 21, might be used by either an end point or a base station in order to disconnect the end point. Analogue to the attach request control segment, the user specific signature is not processed by the link layer but is intended to verify the detachment in the backend.

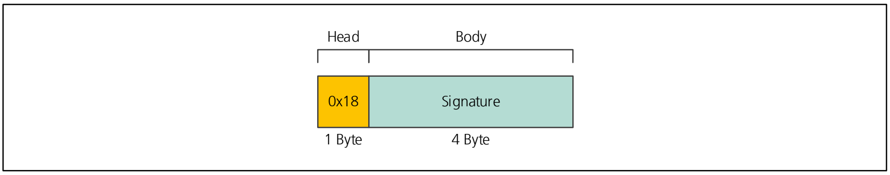


Figure 21: Detach request control segment

Detach accept

Figure 22 depicts the detach accept control segment used as a response to a prior detach request by either the base station or the end point. The contained signature is again intended for backend verification purposes and is not processed within the link layer.

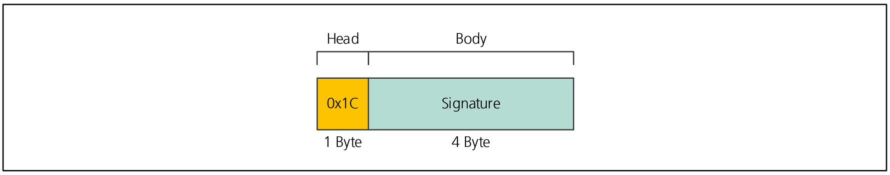


Figure 22: Detach accept control segment

Downlink RX status query

A downlink RX status query can be used by the base station to request information about the downlink reception at an end point. This is important as it is not valid to assume symmetric channel conditions between uplink and downlink under all circumstances. This is especially true if different frequency bands are used in up- and downlink. Thus the base station might require information from the end point itself to properly estimate channel conditions.

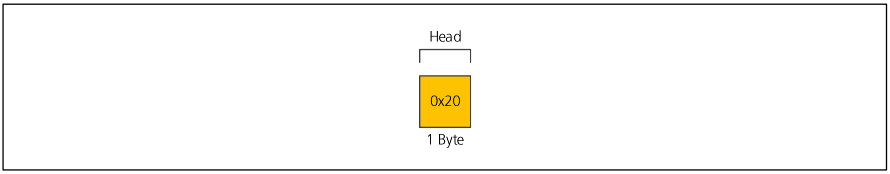


Figure 23: Downlink RX status query control segment

As shown in Figure 23, the downlink RX status query is purely a flag represented by the presence of the header and does not carry any extra body.

Downlink RX status response

The downlink RX status response control segment, depicted in Figure 24, is used by an end point to provide a base station with information about the reception characteristics of downlink transmissions. It is sent after receiving a downlink RX status request by the base station. Especially unidirectional end points might also opt to include a downlink RX status response without prior request in their uplink transmissions at a regular interval.

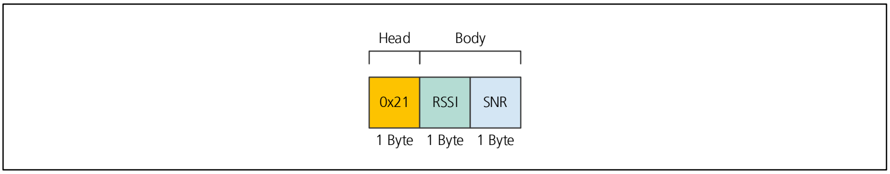


Figure 24: Downlink RX status response control segment

The RSSI field contains a received signal strength indicator as an unsigned fixed point integer with a least significant bit value of 1 dBm and an offset of -174 dBm. Accordingly a value of 0x00 corresponds to -174 dBm and a value of 0xFF corresponds to 81 dBm.

The SNR field provides the signal to noise ratio as a two compliment signed fixed point integer with a least significant bit value of 0.5 dB. So for example a value of 0x10 corresponds to 8 dB and a value of 0xF8 corresponds to -4 dB.

Link adaptation request

A link adaptation request control segment allows the base station to change the radio settings of an end point during an active attachment. The new radio settings only apply after the according confirmation has been received from the end point.

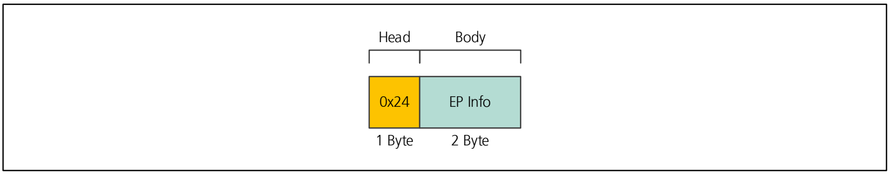


Figure 25: Link adaptation request control segment

As can be seen in Figure 25, again the end point info field described in section 3.2.2 is utilized to select the radio settings for the end point. The requested settings must conform to the end point capabilities advertised in the attachment procedure.

Link adaptation confirm

An end point responds to a link adaptation request by sending a link adaptation confirm control segment. After the confirmation the new radio settings are applied at the base station and the end point, thus it is usually advisable to utilize an acknowledgement for the packet containing the confirmation. Please refer to section 2.5.1 for more information about acknowledgements.

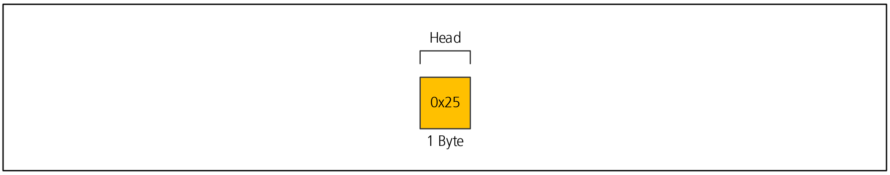


Figure 26: Link adaptation confirm control segment

The link adaptation confirm control segment, depicted in Figure 26, comprises only the header without any attached body fields.

3.2.2 End point info

The end point info bit field is used in various link layer control segments to negotiate radio settings for an end point. This includes the advertisement of the radio capabilities by the end point in the attach request as well as the selection of actual radio settings by the base station in the attach accept or link adaptation request. Please refer to the according descriptions in section 3.2.1 for details about the control segments.

The available radio options and their mappings to the 16 bit end point info field are listed in Table 5. These options mostly correspond to MIOTY™ physical layer modes and settings.

Bit	Function	Direction	Description
0	Channel Use	UL	0: Single channel 1: Dual channel
1	Repetition	UL	0: Repetitions not used 1: Repetitions used
2	Carrier Offset value	UL	0: $n_{co} = 3$ 1: $n_{co} = 11$
3	DL Interblock Distance	DL	0: short ($T_{DN} = 512$) 1: long ($T_{DN} = 7168$)
4	UL-TS-ULP	UL	0: UL-TS-ULP data rate not supported 1: UL-TS-ULP data rate supported
5	UL-TS-ER	UL	0: UL-TS-ER data rate not supported 1: UL-TS-ER data rate supported
6	DL-SB	DL	0: DL-SB Mode not supported 1: DL-SB Mode supported
7	DL-TS-ULP	DL	0: DL-TS mode with ULP data rate not supported 1: DL-TS mode with ULP data rate supported
8-15	Reserved	-	Reserved for future use, set to zero

Table 5: End point info field

A detailed description of the physical layer modes and settings is beyond the scope of this document; please refer to the "MIOTY™ – Physical Layer Technology" document for more information.

4 Backend

The backend in a MIOTY™ system comprises the service center and the application center. While the service center is responsible for network management, and thus is a key component of a MIOTY™ system, the application center is on the edge between MIOTY™ and the application domain, providing the interface for applications to the system. This allows for a strict separation of the MIOTY™ network from application specific concerns. For example the service center might be operated by a network service provider to offer coverage to customers. Each customer might then operate its own application center, keeping the application level cryptography and associated keys on premises. Of course the entire backend might also be managed by one party or might be partially realized as a cloud service depending on the requirements of the network setup.

4.1 Backend communication

Communication between base station, service center and application center relies on TCP/IP connections. This allows for a maximum of flexibility in network configurations as it allows utilization of a private Ethernet, connections over the internet or the integration of cellular network links to name just a few examples. Furthermore established industry standard security mechanisms can be utilized for the backend communication.

The network level cryptography for the MIOTY™ radio link described in section 2.2 is therefore substituted in the backend by TLS secured connections with two sided authentication via certificates as depicted in Figure 27. This allows for a verified identification of every network component like base stations, service centers and application centers and also protects control and payload data with state of the art cryptography. A comprehensive assessment of the security concept for the entire MIOTY™ system can be found in the "MIOTY™ – Security Guideline" document.

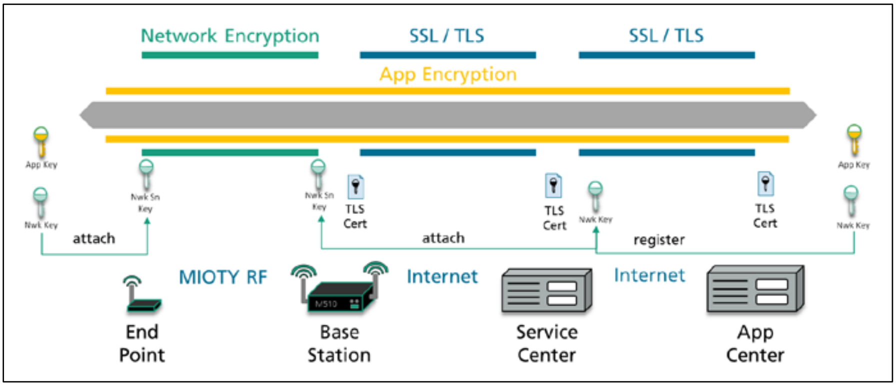


Figure 27: Encryption in the MIOTY™ communication chain

A persistent TCP link to the service center is established and maintained by every base station and by every application center of the system. Therefore the service center must be available at a fixed network location and this network location must be provided to base stations and application centers during system configuration. Base stations and application centers are not required to accept any incoming connections and can thus i.e. be located behind a NAT firewall.

4.2 BSSC interface

The Base Station Service Center Interface (BSSCI) describes the communication formats between a base station and the service center.

The BSSCI protocol is based on JSON [6] objects and MessagePack [7] encoded. Every operation is transferred as a separate object containing the type of the operation and all required meta information. Each JSON/MessagePack object is preceded with a binary header section to allow packetizing of the TCP stream.

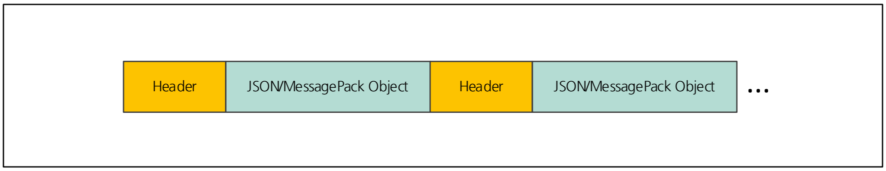


Figure 28: Packetization of the BSSCI data stream

4.2.1 Header

The header includes an identifier and a size field.

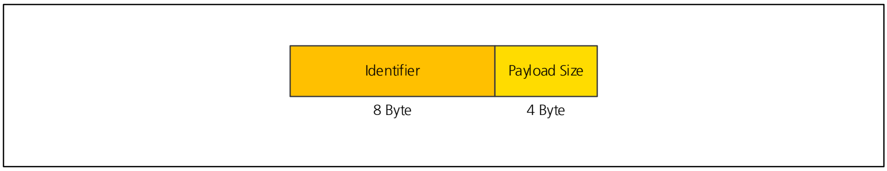


Figure 29: Header format of the BSSCI packets

Field	Description
Identifier	"MIOTYB01" ASCII encoded, 0x 4d 49 4f 54 59 42 30 31
Payload Size	Size of the following JSON/MessagePack object in bytes, little endian

Table 6: Fields and descriptions of the header

4.2.2 Core fields

Core fields are mandatory fields in the JSON object for all BSSCI messages.

Name	Type	Description
version	String	Protocol version, "major.minor.patch", current "1.0.0"
command	String	Type of operation as specified in the following sections
time	Numeric	Unix timestamp as 64bit signed for current UTC time
bsEui	Numeric	EUI64 of the base station of origin or destination

Table 7: Names, types and descriptions of the core fields

4.2.3 Base station messages

The following messages are sent from the base station to the service center.

Base station connect

The base station connect message is sent by the base station immediately after establishing the TCP/IP connection with the service center.

Name	Type	Description
version	String	Protocol version
command	String	"baseCon"
Time	Numeric	Unix timestamp as 64bit signed for current UTC time
bsEui	Numeric	Base station EUI64

Table 8: Names, types and descriptions of the base station connect massage

Base station status

The base station status message is sent by the base station in a defined interval or at certain events. All status fields and the message itself are optional.

Name	Type	Description
version	String	Protocol version
command	String	"baseStatus"
time	Numeric	Unix timestamp as 64bit signed for current UTC time
bsEui	Numeric	Base station EUI64
status	String	"ok" or "error"
uptime	Numeric	System uptime in seconds
temp	Numeric	System temperature in degree Celsius
cpuLoad	Numeric	CPU utilization in percent
memLoad	Numeric	Memory utilization in percent

Table 9: Names, types and descriptions of the base station status

Attach request

The attach request message is sent by the base station after receiving an over the air attachment request from an end point.

Name	Type	Description
version	String	Protocol version
command	String	“attReq”
time	Numeric	Unix timestamp as 64bit signed for current UTC time
bsEui	Numeric	Base station EUI64
epEui	Numeric	End point EUI64
attachCnt	Numeric	End point attachment counter
signalLevel	Numeric	Reception signal level
noiseLevel	Numeric	Reception noise level
nonce	Numeric[4]	4 Byte end point nonce
sign	Numeric[4]	4 Byte end point signature
nwkAddr	Numeric	End point network local short address, as-signed by the base station
dualChan	Boolean	True if end point uses dual channel mode
repetition	Boolean	True if end point uses UL repetition
wideCarrOff	Boolean	True if end point uses wide carrier offset
longBlkDist	Boolean	True if end point uses long DL interblock dis-tance

Table 10: Names, types and descriptions of the attach request message

Detach request

The detach request message is sent by the base station after receiving an over the air detachment request from an end point.

Name	Type	Description
version	String	Protocol version
command	String	“detReq”
time	Numeric	Unix timestamp as 64bit signed for current UTC time
bsEui	Numeric	Base station EUI64
epEui	Numeric	End point EUI64
packetCnt	Numeric	End point packet counter
signalLevel	Numeric	Reception signal level
noiseLevel	Numeric	Reception noise level
sign	Numeric[4]	4 Byte end point signature

Table 11: Names, types and descriptions of the detach request massage

Attach propagate accept

The attach propagate accept message is sent by the base station after receiving an attachment propagate message from the service center.

Name	Type	Description
version	String	Protocol version
command	String	"attPrpAcc"
time	Numeric	Unix timestamp as 64bit signed for current UTC time
bsEui	Numeric	Base station EUI64
epEui	Numeric	End point EUI64

Table 12: Names, types and descriptions of the attach propagate accept message

Detach propagate accept

The detach propagate accept message is sent by the base station after receiving a detachment propagate message from the service center.

Name	Type	Description
version	String	Protocol version
command	String	"detPrpAcc"
time	Numeric	Unix timestamp as 64bit signed for current UTC time
bsEui	Numeric	Base station EUI64
epEui	Numeric	End point EUI64

Table 13: Names, types and descriptions of the detach propagate accept message

DL RX status response

The downlink RX status response message is sent by the base station after receiving a DL RX status response control segment from an end point.

Name	Type	Description
version	String	Protocol version
command	String	"dlRxStatRes"
time	Numeric	Unix timestamp as 64bit signed for current UTC time

bsEui	Numeric	Base station EUI64
epEui	Numeric	End point EUI64
dlRxSignalLevel	Numeric	End point DL reception signal level
dlRxNoiseLevel	Numeric	End point DL reception noise level

Table 14: Name, types and descriptions of the downlink RX status response message

RX data

The RX data message is sent by the base station after receiving uplink data from an end point.

Name	Type	Description
version	String	Protocol version
command	String	"rxData"
time	Numeric	Unix timestamp as 64bit signed for current UTC time
bsEui	Numeric	Base station EUI64
epEui	Numeric	End point EUI64
packetCnt	Numeric	End point packet counter
signalLevel	Numeric	Reception signal level
noiseLevel	Numeric	Reception noise level
userData	Numeric[n]	n Byte End point user data
rxOpen	Boolean	True if End point downlink window is opened

Table 15: Name, types and descriptions of the RX data massage

TX data result

The TX data result message is sent by the base station when queued TX data has either been sent or discarded.

Name	Type	Description
version	String	Protocol version
command	String	"txDataRes"
time	Numeric	Unix timestamp as 64bit signed for current UTC time
bsEui	Numeric	Base station EUI64
epEui	Numeric	End point EUI64

packetCnt	Numeric	End point packet counter of the scheduled data
result	String	“sent”, “expired”, “invalid”, …

Table 16: Names, types and descriptions of the TX data result message

4.2.4 Service center messages

The following messages are sent from the service center to the base station.

Attach request accept

The attach request accept message is sent by the service center after receiving an attachment request message from the base station.

Name	Type	Description
version	String	Protocol version
command	String	“attReqAcc”
time	Numeric	Unix timestamp as 64bit signed for current UTC time
bsEui	Numeric	Base station EUI64
epEui	Numeric	End point EUI64
nwkSnKey	Numeric[16]	16 Byte end point network session key

Table 17: Names, types and descriptions of the attach request accept massage

Detach request accept

The detach request accept message is sent by the service center after receiving a detachment request message from the base station.

Name	Type	Description
version	String	Protocol version
command	String	“detReqAcc”
time	Numeric	Unix timestamp as 64bit signed for current UTC time
bsEui	Numeric	Base station EUI64
epEui	Numeric	End point EUI64
sign	Numeric[4]	4 Byte signature for the end point

Table 18: Names, types and descriptions of the detach request accept massage

Attach propagate

The attach propagate message is sent by the service center to propagate an end point attachment to the base station. The attachment information can either be acquired via an over the air attach at another base station or in the form of an offline preattachment of an end point (as required for unidirectional end points).

Name	Type	Description
version	String	Protocol version
command	String	"attPrp"
time	Numeric	Unix timestamp as 64bit signed for current UTC time
bsEui	Numeric	Base station EUI64
epEui	Numeric	End point EUI64
bidi	Boolean	True if end point is bi-directional
nwkSnKey	Numeric[16]	16 Byte end point network session key
nwkAddr	Numeric	End point network local short address
lastPacketCnt	Numeric	Last known end point packet counter
dualChan	Boolean	True if end point uses dual channel mode
repetition	Boolean	True if end point uses UL repetition
wideCarrOff	Boolean	True if end point uses wide carrier offset
longBlkDist	Boolean	True if end point uses long DL interblock distance

Table 19: Names, types and descriptions of the attach propagate message

Detach propagate

The detach propagate message is sent by the service center to propagate an end point detachment to the base station.

Name	Type	Description
version	String	Protocol version
command	String	"detPrp"
time	Numeric	Unix timestamp as 64bit signed for current UTC time

bsEui	Numeric	Base station EUI64
epEui	Numeric	End point EUI64

Table 20: Names, types and descriptions of the detach propagate message

DL RX status query

The downlink RX status query message is sent by the service center to schedule a DL RX status query control segment for the next downlink transmission of the base station to an end point.

Name	Type	Description
version	String	Protocol version
command	String	"dlRxStatQry"
time	Numeric	Unix timestamp as 64bit signed for current UTC time
bsEui	Numeric	Base station EUI64
epEui	Numeric	End point EUI64

Table 21: Names, types and descriptions of the downlink RX status query message

TX data queue

The TX data queue message is sent by the service center to schedule downlink data in the base station for an end point. This might be done either within the pause between an uplink message and the according downlink window for direct responses or a priori for predefined downlink data. Scheduled downlink data which is counter dependent can only be transmitted in a downlink window with the matching counter.

Name	Type	Description
version	String	Protocol version
command	String	"txDataQue"
time	Numeric	Unix timestamp as 64bit signed for current UTC time
bsEui	Numeric	Base station EUI64
epEui	Numeric	End point EUI64
packetCnt	Numeric	End point packet counter, must match RX packet counter
userData	Numeric[n]	n Byte end point user data

cntDepend	Boolean	True if userData is counter dependent (encryption etc.)
responseExp	Boolean	True if end point response is expected, optional
responsePrio	Boolean	True for priority end point response, optional
dlWindReq	Boolean	True to request further end point DL window, optional

Table 22: Names, types and descriptions of the TX data queue message

TX data revoke

The TX data revoke message is sent by the service center to revoke previously scheduled downlink data at the base station.

Name	Type	Description
version	String	Protocol version
command	String	"txDataRev"
time	Numeric	Unix timestamp as 64bit signed for current UTC time
bsEui	Numeric	Base station EUI64
epEui	Numeric	End point EUI64
packetCnt	Numeric	End point packet counter of the scheduled data

Table 23: Names, types and descriptions of the TX data revoke message

4.3 SCAC interface

The Service Center Application Center Interface (SCACI) is used for communication between a service center and an application center.

The SCACI protocol is based on JSON [6] objects and MessagePack [7] encoded. Every operation is transferred as a separate object containing the type of the operation and all required meta information. Each JSON/MessagePack object is preceded with a binary header section to allow packetizing of the TCP stream.

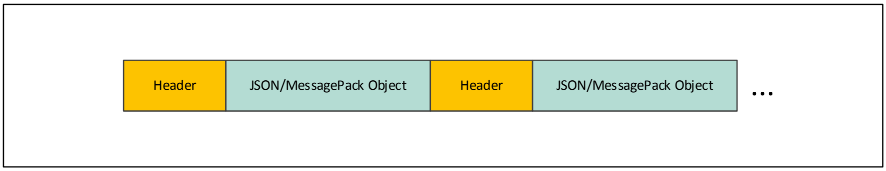


Figure 30: Packetization of the SCACI data stream

4.3.1 Header

The header includes an identifier and a size field.

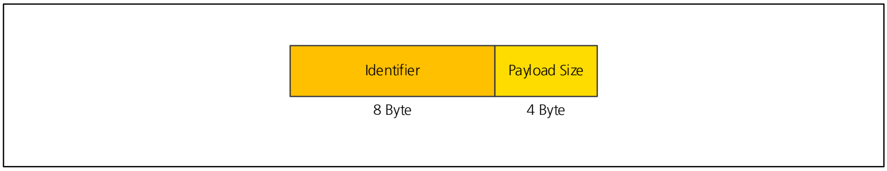


Figure 31: Header format of the SCACI packets

Field	Description
Identifier	"MIOTYA01" ASCII encoded, 0x 4d 49 4f 54 59 41 30 31
Payload Size	Size of the following JSON/MessagePack object in bytes, little endian

Table 24: Fields and descriptions of the header

4.3.2 Core fields

Core fields are mandatory fields in the JSON object for all SCACI messages.

Name	Type	Description
version	String	Protocol version, "major.minor.patch", current "1.0.0"
command	String	Type of operation as specified in the following sections
time	Numeric	Unix timestamp as 64bit signed for current UTC time
acEui	Numeric	EUI64 of the application center of origin or destination

Table 25: Names, types and descriptions of the core fields

4.3.3 Application center messages

The following messages are sent from the application to the service center.

Application center connect

The application center connect message is sent by the application center immediately after establishing the TCP/IP connection with the service center.

Name	Type	Description
version	String	Protocol version
command	String	"appCenterCon"
time	Numeric	Unix timestamp as 64bit signed for current UTC time
acEui	Numeric	Application center EUI64

Table 26: Names, types and description of the application center connect

Register request

The register request message is sent by the application center to register an end point at the service center.

Name	Type	Description
version	String	Protocol version
command	String	"regReq"
time	Numeric	Unix timestamp as 64bit signed for current UTC time
acEui	Numeric	Application center EUI64
epEui	Numeric	End point EUI64
bidi	Boolean	True for bi-directional end points
nwkKey	Numeric[16]	16 Byte end point network key
nwkAddr	Numeric	End point network local short address
lastAttachCnt	Numeric	Last known end point attachment counter
lastPacketCnt	Numeric	Last known end point packet counter
dualChan	Boolean	True if end point uses dual channel mode
repetition	Boolean	True if end point uses UL repetition
wideCarrOff	Boolean	True if end point uses wide carrier offset
longBlkDist	Boolean	True if end point uses long DL interblock distance

Table 27: Names, types and description of the register request message

Deregister request

The deregister request message is sent by the application center to deregister an end point at the service center.

Name	Type	Description
version	String	Protocol version
command	String	"deregReq"
time	Numeric	Unix timestamp as 64bit signed for current UTC time
acEui	Numeric	Application center EUI64
epEui	Numeric	End point EUI64

Table 28: Names, types and descriptions of the deregister request

TX data queue

The TX data queue message is sent by the application center to schedule downlink data in the service center for an end point. This might be done either within the pause between an uplink message and the according downlink window for direct responses or a priori for predefined downlink data. Scheduled downlink data which is counter dependent can only be transmitted in a downlink window with the matching counter.

Name	Type	Description
version	String	Protocol version
command	String	"txDataQue"
time	Numeric	Unix timestamp as 64bit signed for current UTC time
acEui	Numeric	Application center EUI64
epEui	Numeric	End point EUI64
packetCnt	Numeric	End point packet counter, must match RX packet counter
userData	Numeric[n]	n Byte end point user data
cntDepend	Boolean	True if userData is counter dependent (encryption etc.)
responseExp	Boolean	True if end point response is expected, optional
responsePrio	Boolean	True for priority end point response, optional

dlWindReq	Boolean	True to request further end point DL window, optional

Table 29: Names, types and descriptions of the TX data queue

TX data revoke

The TX data revoke message is sent by the application center to revoke previously scheduled downlink data at the service center.

Name	Type	Description
version	String	Protocol version
command	String	"txDataRev"
time	Numeric	Unix timestamp as 64bit signed for current UTC time
acEui	Numeric	Application center EUI64
epEui	Numeric	End point EUI64
packetCnt	Numeric	End point packet counter of the scheduled data

Table 30: Names, types and descriptions of the TX data revoke message

4.3.4 Service center messages

The following messages are sent from the service center to the application center.

Register request accept

The register request accept message is sent by the service center after receiving a registration request message from the application center.

Name	Type	Description
version	String	Protocol version
command	String	"regReqAcc"
time	Numeric	Unix timestamp as 64bit signed for current UTC time
acEui	Numeric	Application center EUI64
epEui	Numeric	End point EUI64

Table 31: Names, types and descriptions of the register request accept

Deregister request accept

The deregister request accept message is sent by the service center after receiving a deregistration request message from the application center.

Name	Type	Description
version	String	Protocol version
command	String	"deregReqAcc"
time	Numeric	Unix timestamp as 64bit signed for current UTC time
acEui	Numeric	Application center EUI64
epEui	Numeric	End point EUI64

Table 32: Names, types and descriptions of the deregister request accept message

End point status update

The end point status update message is sent by the service center when an end point is attached or detached from the system.

Name	Type	Description
version	String	Protocol version
command	String	"epStatUpd"
time	Numeric	Unix timestamp as 64bit signed for current UTC time
acEui	Numeric	Application center EUI64
epEui	Numeric	End point EUI64
epStatus	String	"attached" or "detached"
attachCnt	Numeric	End point attachment counter, over the air attach only

Table 33: Names, types and descriptions of the end point status update message

RX data

The RX data message is sent by the service center after receiving uplink data from an end point via a base station.

Name	Type	Description
version	String	Protocol version
command	String	"rxData"
time	Numeric	Unix timestamp as 64bit signed for current UTC time
acEui	Numeric	Application center EUI64
bsEui	Numeric	Base station EUI64
epEui	Numeric	End point EUI64
packetCnt	Numeric	End point packet counter
signalLevel	Numeric	Reception signal level
noiseLevel	Numeric	Reception noise level
userData	Numeric[n]	n Byte End point user data
rxOpen	Boolean	True if End point downlink window is opened

Table 34: Names, types and descriptions of the RX data message

TX data result

The TX data result message is sent by the base station when queued TX data has either been sent or discarded.

Name	Type	Description
version	String	Protocol version
command	String	"txDataRes"
time	Numeric	Unix timestamp as 64bit signed for current UTC time
acEui	Numeric	Application center EUI64
bsEui	Numeric	Base station EUI64
epEui	Numeric	End point EUI64
packetCnt	Numeric	End point packet counter of the scheduled data
result	String	"sent", "expired", "invalid", …

Table 35: Names, types and descriptions of the TX data result

References

[1] Publication 197 (2001): "Specification for the Advanced Encryption Standard (AES)", NIST Processing Standards".
http://nvlpubs.nist.gov/nistpubs/FIPS/NIST.FIPS.197.pdf

[2] Recommendation for Block Cipher Modes of Operation (Methods and Techniques), Morris Dworkin, NIST Special Publication 800-38A, Edition 2001.
http://nvlpubs.nist.gov/nistpubs/Legacy/SP/nistspecialpublication800-38A.pdf

[3] NIST Special Publication 800-38B (2005): "Recommendation for Block Cipher Modes of Operation: the CMAC Mode for Authentication".
https://csrc.nist.gov/publications/detail/sp/800-38b/final

[4] IETF RFC 4493: "The AES-CMAC Algorithm", 2006.
https://tools.ietf.org/html/rfc4493

[5] IEEE™ Guidelines for Use of Extended Unique Identifier (EUI), Organizationally Unique Identifier (OUI), and Company ID (CID).
http://standards.ieee.org/develop/regauth/tut/eui.pdf

[6] ECMA-404: "The JSON Data Interchange Syntax", 2nd Edition, 2017.
http://www.ecma-international.org/publications/files/ECMA-ST/ECMA-404.pdf

[7] MessagePack specification, 2017.
https://github.com/msgpack/msgpack/blob/master/spec.md